AutoXAI: A Framework to Automatically Select the Most Adapted XAI Solution

Robin Cugny
SolutionData Group
Université Toulouse 2, IRIT
Toulouse, France
robin.cugny@irit.fr

Julien Aligon
Université Toulouse 1, IRIT
Toulouse, France
julien.aligon@irit.fr

Max Chevalier
Université Toulouse 3, IRIT
Toulouse, France
max.chevalier@irit.fr

Geoffrey Roman Jimenez
SolutionData Group
Toulouse, France
groman-jimenez@solutiondatagroup.fr

Olivier Teste
Université Toulouse 2, IRIT
Toulouse, France
olivier.teste@irit.fr

ABSTRACT

A large number of XAI (eXplainable Artificial Intelligence) solutions have been proposed in recent years. Recently, thanks to new XAI evaluation metrics, it has become possible to compare these XAI solutions. However, selecting the most relevant XAI solution among all this diversity is still a tedious task, especially if a user has specific needs and constraints. In this paper, we propose AutoXAI, a framework that recommends the best XAI solution and its hyperparameters according to specified XAI evaluation metrics while considering the user's context (dataset, machine learning model, XAI needs and constraints). It adapts approaches from context-aware recommender systems on one side and strategies of optimization and evaluation from AutoML (Automated Machine Learning) on the other. Through two use cases, we show that AutoXAI recommends XAI solutions adapted to the user's needs with the best hyperparameters matching the user's constraints.

CCS CONCEPTS

• **Computing methodologies → Machine learning**.

KEYWORDS

Explainable machine learning, Evaluation of explainability, Quality of explanation, Evaluation metrics, AutoML, Recommender system, Information system

ACM Reference Format:
Robin Cugny, Julien Aligon, Max Chevalier, Geoffrey Roman Jimenez, and Olivier Teste. 2022. AutoXAI: A Framework to Automatically Select the Most Adapted XAI Solution. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557247

ACKNOWLEDGMENTS

This work was supported by ANRT (CIFRE [2020/0870]) in collaboration with SolutionData Group and IRIT.

1 INTRODUCTION

Machine Learning (ML) models are now widely used in the industry. However, their lack of understandability delays their adoption in high stakes domains such as the medical field [1], digital security [2], judicial field [3], or autonomous driving [4]. In such contexts, decision-makers should understand ML models and their results to detect biases [3] or meaningless relationships [5]. During the last decade, the eXplainable Artificial Intelligence (XAI) field proposed a wide variety of solutions to facilitate the understanding of ML models [6–12]. In view of the growing number of XAI proposals [9], evaluating the quality of explanations has become necessary to choose an appropriate XAI solution as well as its hyperparameters. It is worth noting that the evaluation of explanations can either be done subjectively by humans or objectively with metrics [13–15]. However, data scientists who want to include an XAI solution have the following issues:

- They must check which XAI solutions are compatible with the data type and the ML model.
- The XAI solutions should explain specifically what the data scientists want to understand and it should be explained in an appropriate format.
- They should evaluate the effectiveness of the explanations produced by the selected XAI solutions.
- The context requires that the explanations match specific quality criteria (called explanations' properties) which imposes the use of appropriate evaluation metrics.
- They have to find the best hyperparameters for each of the selected XAI solutions to keep the best of them.

These are tedious and time-consuming tasks. Theoretical guides have been proposed by [16, 17] but, to the best of our knowledge, automating the complete XAI recommendation approach has never been formalized and implemented before.

In this paper, we propose to automate these tasks in a framework to assist data scientists in choosing the best XAI solutions according to their context (dataset, ML model, XAI needs and constraints).

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557247

Suggesting an adapted XAI solution requires defining the elements of the data scientists' contexts and using them to filter the compatible XAI solutions. This task is challenging because there are as many formalizations as there are authors in the XAI field and very few works have attempted to unify XAI elements with a formalization [15, 17, 18]. As we want to evaluate the XAI solutions, we must automatically find the XAI evaluation metrics that are compatible with the XAI solutions and are meaningful to the context. Moreover, it is necessary to find a way to validate multiple complementary properties by optimizing corresponding XAI evaluation metrics while considering the data scientists' preferences. Indeed, properties' importance is subjective and depend on the context. In addition, ranking XAI solutions requires finding the best hyperparameters using XAI evaluation metrics. As it is computationally expensive, we draw inspiration from time-saving strategies for model evaluation in AutoML [19].

The contributions of this paper are as follows:

- AutoXAI, a framework that recommends XAI solutions to match the data scientists' context and optimize their hyperparameters with respect to XAI evaluation metrics [14, 15].
- A more generic formalization of the data scientists' context for XAI.
- A new evaluation metric to assess the completeness of example-based explanations.
- New time-saving strategies adapted to XAI evaluation.

We illustrate AutoXAI's recommendations through two use cases with different users' constraints and needs as well as different datasets and models. These studies let us uncover interactions between hyperparameters and properties of explanations, as well as interactions between the properties themselves.

The rest of the paper is organized as follows. Related work are described in Section 2. Formal definitions are illustrated by an example of context in Section 3. The core of our framework is detailed in Sections 4 and 5. Experiments of Section 6 show that our framework is well adapted to propose an explanation matching the user's context and that time-saving strategies considerably reduce the computation time. Finally, we conclude the paper and give possible perspectives in Section 7.

2 RELATED WORK

Four research topics interact in this paper: the XAI solutions, which are being assessed by XAI evaluation metrics while context-aware recommender systems and AutoML are means used to propose the most adapted solution.

2.1 XAI solutions

In this paper, we define an XAI solution as any algorithm that produces an explanation related to an ML problem. This includes methods that explain black-box models but also naturally interpretable models. As mentioned in Section 1, many XAI solutions now exist and different taxonomies have been proposed such as [6, 7, 9, 12, 15]. [7] also suggests grouping XAI solutions according to the type of explanation produced. They list: feature summary, model internals, data point, surrogate intrinsically interpretable model, rule sets, explanations in natural language, and question-answering. Later, [20] suggests that XAI explanations answer specific questions about data, its processing and results in ML. They map existing XAI solutions to questions and create an XAI question bank that supports the design of user-centered XAI applications. [21] defines an explanation as an explanan: the answer to the question and an explanandum: what is to be explained. These two elements provide a user-friendly characterization of explanations and thus allow the user to specify which explanation is more adapted.

The diversity of existing XAI solutions makes it hard to find an XAI solution adapted to one's needs. Moreover, as the XAI field is growing, more and more XAI solutions proposed in the literature are producing similar kinds of explanations. Hence, it has become necessary to objectively compare XAI solutions by assessing the effectiveness of their explanations. In this direction, the recent literature has focused on quantitative XAI evaluations [15].

2.2 Evaluation of XAI solutions

[10] distinguishes three strategies of evaluation: application-grounded evaluation, human-grounded evaluation, and functionality-grounded evaluation that does not imply human intervention. Application-grounded evaluation tests the effectiveness of explanations in a real-world application with domain experts and human-grounded evaluation are carried out with lay humans. While explanations are intended for humans, functionality-grounded evaluations are interesting because of their objectivity. Thus, this type of evaluation is inexpensive, fast and can lead to a formal comparison of explanation methods [14].

Since the notion of "good explanations" is not trivial, some quality properties have been proposed by [22]. These are man-made criteria that attest to the quality of the explanations. Functionality-grounded evaluation metrics are constructed to calculate scores to measure how well a property is met.

[15] focuses on the functionality-grounded evaluation and proposed the Co-12 Explanation Quality Properties to unify the diverse properties proposed in the literature. They reviewed most XAI evaluation metrics and associate each of them with properties. Examples of their properties that will be studied in this paper are as follow: *Continuity* describes how continuous and generalizable the explanation function is, *Correctness* describes how faithful the explanation is w.r.t. the black box, *Compactness* describes the size of the explanation, and *Completeness* describes how much of the black box behavior is described in the explanation.

In practice, XAI evaluation metrics produce scores for properties of interest, making it possible to compare and choose an XAI solution. However, the data scientists still have to find the desired XAI solutions and their corresponding XAI evaluation metrics. This issue could be addressed with strategies that have been studied in context-aware recommender systems.

2.3 Context-aware recommender systems

Recommender systems filter information to present the most relevant elements to a user. To the best of our knowledge, there is no recommender system for XAI solutions. To recommend adapted XAI solutions, one should consider the whole context of the data scientist. According to [23], context-aware recommender systems offer more relevant recommendations by adapting them to the user's situation. They also state that context may be integrated during three phases: contextual prefiltering which selects a subset of possible candidates before the recommendation, contextual modeling which uses context in the recommendation process, and contextual postfiltering which adjusts the recommendation afterward. These three phases require formally defining the elements of the context, which is one of our objectives for the framework we propose in this paper. While recommending an adapted XAI solution is a first interesting step, the data scientist eventually wants a reliable explanation, i.e. an explanation that verifies the properties of interest. To achieve this, a possible approach is to use previously detailed XAI evaluation metrics to optimize hyperparameters of adapted XAI solutions. For this kind of approach, many strategies have been proposed in the AutoML domain.

2.4 AutoML

Designing ML algorithms is an iterative task of testing and modifying both the architecture and the hyperparameters of the algorithm. It is a repetitive task that requires a lot of time. For this reason, a part of the research has focused on automating the design of ML algorithms, namely AutoML [19]. AutoML frameworks look for the best performing ML pipeline treatment to solve a task on a given dataset. According to [19], AutoML consists of several processes: data preparation, feature engineering, model generation, and model evaluation. They divide the model generation process into two steps: search space and optimization methods. The first step defines the design principles of models that are tested, and the second is how to obtain the best scores in the model evaluation process. The main strategy of interest here is HyperParameter Optimization (HPO) which consists in finding the best hyperparameters according to a loss function. As model performances cannot be derived according to a hyperparameter, it is a non-differentiable optimization problem, therefore the HPO methods do not rely on the model to propose a solution. Moreover, since training a model until convergence is a costly operation, model evaluation is a very time-consuming step. Thus, several strategies have been proposed to accelerate the evaluation of models. [19] lists four types of strategies: *low fidelity*, *weight sharing*, *surrogate*, and *early stopping*. The *low fidelity* strategy consists in reducing the number of observations to reduce the number of epochs, or the size of observations to reduce the number of parameters to optimize. *Weight sharing* reuses learned parameters between models. The *surrogate* strategy replaces a computationally expensive model with an approximation to estimate the performance of neural architecture and guide architecture research. *Early stopping* can accelerate model evaluation by stopping iterations if performances are predicted to be lower than the current best score.

2.5 Approaches for choosing an XAI solution

As mentioned in Section 1, to select an XAI solution, the data scientist can currently rely on XAI libraries, benchmarks, and AutoML frameworks. Currently, available XAI libraries such as DeepExplain [24], AI Explainability 360 [25], and Alibi [26] are gathering state-of-the-art XAI solutions. However, they neither integrate automatic evaluation of explanation nor recommend XAI solutions according to data scientists' needs and constraints.

Comparatives and benchmarks [27–29] compare XAI solutions efficiency using XAI evaluation metrics. They are often joint with the proposal of an XAI solution or an XAI evaluation metric on which they are focusing. However, the results obtained depend on the dataset and the ML model that may not be the ones the data scientist uses, and thus the results may be different. Moreover, the hyperparameters of the XAI solutions are not optimized to maximize the various properties needed by the data scientist. This last point is problematic as some XAI properties such as correctness and compactness are not independent [15].

Eventually, [16] highlights that users should be guided in choosing XAI solutions and proposes a methodology for this issue, while [17] proposes a theoretical framework to facilitate the comparison between XAI solutions.

To summarize, as there is a high diversity of XAI solutions, it is a complex and tedious task for data scientists to find XAI solutions that fit their needs. Yet, there is no recommendation system to automate this task. Moreover, data scientists look for the best XAI solution as they want a reliable solution. XAI evaluation metrics allow for objective comparison, but XAI libraries do not implement them and comparatives are not adapted to the user's context. Eventually, data scientists have to find hyperparameters to maximize the desired properties. However, this task should be done for multiple XAI solutions and multiple properties according to the data scientists' preferences. This paper aims to address these issues in the following sections.

3 EXAMPLE AND FORMALIZATION

3.1 Illustrative example

Let's first consider a data scientist in a medical laboratory, Alice and Bob, a physician colleague. Bob uses a ML black box model as a decision support tool and asks if it is possible to have an explanation for the predictions of the model to check some rare cases. Alice has access to the model, as well as the data that were used to build it, and now wants to implement an XAI solution to produce explanations.

Here, the needs of Bob, the physician, are the following: the explanations must focus on predictions (as it is asked why they are obtained) and the XAI solution must explain a trained model without modifying it. Moreover, Bob wants to know how the collected data for a patient (the features) influence the result of the model.

Regarding the constraints of the context, the high-stakes decisions impose the use of a precise model and the most faithful explanations possible (correctness property). Nevertheless, the explanations should not be completely changed by small perturbations as blood measurements might be noisy, therefore, stable explanations are mandatory (continuity property). Eventually, since Bob will be the main user of these explanations, concise explanations should be encouraged as physicians shouldn't waste time on unimportant features (compactness property).

3.2 Definitions

Definition 3.1 (Dataset). Let X, Y be a dataset with $X = \{x_i\}_{i=1}^{n} | x_i \in \mathbb{R}^d$ the observations and $Y = \{y_i\}_{i=1}^{n} | y_i \in \mathbb{R}$ their corresponding labels, n is the number of observations and d is the number of dimensions of the dataset (also called features).

In our illustrative example, the diabetes dataset [30] X is composed of $n = 442$ patients with $d = 10$ features (physiological information) and Y is the disease progression for each patient.

Definition 3.2 (ML model). A ML model is trained on a dataset X, Y by inferring statistical relationships between X and Y. This model can then be used as predictive function which we note $f : X \rightarrow \hat{Y}$ with $\hat{Y} = \{\hat{y}_i\}_{i=1}^{n} | \hat{y}_i \in \mathbb{R}$, the produced predictions.

In our illustrative example, a Multilayer Perceptron learns a predictive function f that predicts the disease progressions $\hat{Y}$ based on the observations X and the labels Y.

Definition 3.3 (Explanandum and explanan). We note $\mathcal{E} = \{\mathcal{E}_i\}_{i=1}^{k}$ the set of all possible explanandum, where the explanandum $\mathcal{E}_i$ is a descriptor for explanation functions that specifies *what is explained.* We also note $\mathcal{E}' = \{\mathcal{E}'_j\}_{j=1}^{k'}$ the set of all possible explanan, where the explanan $\mathcal{E}'_j$ is a descriptor for explanation functions that specifies *how it is explained.*

In our illustrative example, $\mathcal{E}_i$ *=Why this prediction?* and $\mathcal{E}'_j$ *=feature summaries.*

Definition 3.4 (XAI properties). XAI properties are descriptive quality criteria for explanations. We note P_r, the set properties that explanations verify or not.

In our illustrative example, the properties of interest are correctness, continuity, and compactness.

Thus, in AutoXAI the data scientist can specify the needs with $(\mathcal{E}, \mathcal{E}')$ and constraints with P_r.

Definition 3.5 (XAI solution). An XAI solution acts as a function that produces one or several explanations. We note $E = \{e_t\}_{t=1}^{l}$, the explanations set with $l \in \mathbb{N}$ the number of explanations. We note $f_e^{(h)} : P(X, Y, F, \hat{Y}) \rightarrow E$ the explanation function with $P(X, Y, F, \hat{Y})$ a partition of $\{X, Y, F, \hat{Y}\}$ and h the hyperparameters of the XAI solution. $f_e^{(h)} \in F_e$ with F_e the set of explanation functions. The hyperparameters refer to the static parameters that determine the behaviors of the XAI solution. For naturally interpretable models $f = f_e^{(h)}$.

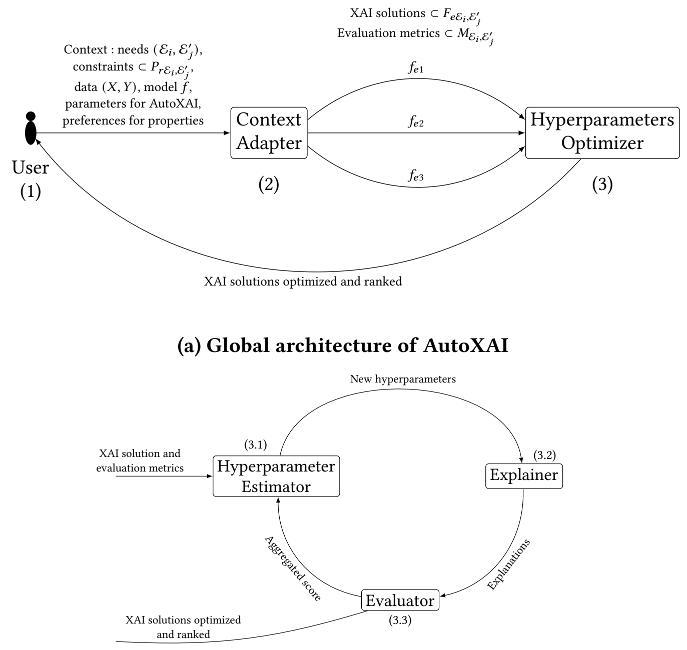
In our illustrative example, the XAI solutions LIME [5] and Kernel SHAP [18] are considered as $f_e^{(h)} : (X, F) \rightarrow E$. Both produce one feature importance explanation $e_t \in \mathbb{R}^d$ for each patient so that $E = \{e_t\}_{t=1}^{n}$.

Definition 3.6 (XAI evaluation metrics). An XAI evaluation metric evaluates one property and is often adapted to one specific type of explanation. We note the set of XAI evaluation metrics $M = \{m_q\}_{q=1}^{c}$, where $m_q : P(X, F, F_e, Y) \rightarrow \mathbb{R}$, with $P(X, F, F_e, Y)$ a partition of $\{X, F, F_e, Y\}$, so that m_q evaluates $p_q \in P_r$.

In our illustrative example, robustness $m_q : (X, F_e) \rightarrow \mathbb{R}$ is an evaluation metric that assess the property of continuity p_q.

4 OUR FRAMEWORK PROPOSAL

We first describe a global step-by-step AutoXAI process with its components as shown in Figure 1a, then we detail the Hyperparameters Optimizer in Figure 1b and eventually each component using the definitions of Section 3.2.


XAI solutions $\subset F_{e\mathcal{E}_i,\mathcal{E}'_j}$
Evaluation metrics $\subset M_{\mathcal{E}_i,\mathcal{E}'_j}$
Context : needs $(\mathcal{E}_i, \mathcal{E}'_j)$,
constraints $\subset P_{r\mathcal{E}_i,\mathcal{E}'_j}$,
data (X, Y), model f,
parameters for AutoXAI,
preferences for properties
f_{e1}
f_{e2}
f_{e3}
Context Adapter
Hyperparameters Optimizer
User
(1)
(2)
(3)
XAI solutions optimized and ranked

(a) Global architecture of AutoXAI

New hyperparameters
XAI solution and evaluation metrics
(3.1)
Hyperparameter Estimator
(3.2)
Explainer
Aggregated score
Explanations
Evaluator
(3.3)
XAI solutions optimized and ranked

(b) Details on Hyperparameters Optimizer

Figure 1: Architecture of AutoXAI.
The figures are read by following the number of the steps. In Figure 1a, for each XAI solution, step (3) optimizes its hyperparameters with respect to the aggregated scores of the evaluation metrics by entering the loop in Figure 1b.

Here are the operations as they are performed, starting with Figure 1a:

1. The User gives the elements of the context, the parameters for AutoXAI and its preferences regarding the properties.
2. The Context Adapter component selects a subset of XAI solutions matching the needs and a subset of evaluation metrics to ensure that the constraints are met.
3. For each XAI solution, the Hyperparameters Optimizer looks for hyperparameters that will reduce the loss function based on the aggregated scores of the evaluation metrics. To do so, it performs the following operations in a loop, see Figure 1b.

3.1. The Hyperparameters Estimator proposes new hyperparameters according to the chosen optimization algorithm.
3.2. The Explainer uses the XAI solution and the newly proposed hyperparameters to produce explanations.
3.3. The Evaluator applies the evaluation metrics to the explanations and aggregates the scores thus obtained.

4.1 Context adapter

As detailed in Section 3.2, XAI solutions can be grouped according to $(\mathcal{E}_i, \mathcal{E}'_j)$, their explanandum and their explanan. This grouping also determines $P_{r\mathcal{E}_i,\mathcal{E}'_j}$, the properties that can describe the XAI solutions, and therefore $M_{\mathcal{E}_i,\mathcal{E}'_j}$, the XAI evaluation metrics that can be applied. To get $(\mathcal{E}_i, \mathcal{E}'_j)$, we ask what the user wants in natural language with pre-written answers. To do so, AutoXAI uses the question bank from [20] for explanandum proposals, letting the user choose which question the XAI solution should answer. For the explanan, AutoXAI uses the list of explanation types from [7]. With this knowledge, a contextual prefiltering (see Section 2.3) is possible by selecting $F_{e\mathcal{E}_i,\mathcal{E}'_j}$ in F_e, $P_{r\mathcal{E}_i,\mathcal{E}'_j}$ in P_r and $M_{\mathcal{E}_i,\mathcal{E}'_j}$ in M. This is possible by tagging each of these proposals with a tuple from $(\mathcal{E},\mathcal{E}')$ using [15, 20] correspondence tables. $P_{r\mathcal{E}_i,\mathcal{E}'_j}$ serve for contextual modeling (see Section 2.3). Indeed, the user can choose the weights (degree of importance) for each of the properties in $P_{r\mathcal{E}_i,\mathcal{E}'_j}$. These weights are used in the evaluation of the XAI solutions which guide the optimization and therefore the ranking of XAI solutions that are produced. 1 is the default value, increasing a weight w_q makes its property p_q more important and conversely, 0 means that the property is ignored. In addition to these elements, AutoXAI can retrieve the model and the dataset on which it was trained if necessary.

4.2 Hyperparameters estimator

This component's goal is to propose new hyperparameters to obtain the best aggregated score. The optimization algorithms are iterative and some, such as Bayesian optimization [31], associate the hyperparameters to a score for building a probabilistic model. This probabilistic model estimates the hyperparameters that should give the best score. Here, f_e, with previously estimated hyperparameters, is evaluated with the aggregated scores of $M_{\mathcal{E}_i,\mathcal{E}'_j}$ (further detailed in Section 4.4) and it results in a new entry to update the probabilistic model. This component estimates hyperparameters according to previous score results in case there are. Otherwise, the values of the hyperparameters are set according to the initialization of the chosen algorithm, for example at random.

4.3 Explainer component

This component's objective is to produce explanations using the XAI solution and the defined hyperparameters. As the implementations of XAI solutions vary in their programming paradigm and in the data structure they use and return, it is necessary to set up a wrapper that standardizes the input and output for each XAI solution of a given group. This component serves as a base to include all implemented XAI solutions and is made to be completed with new XAI solutions.

4.4 Evaluator component

This component aims at computing the scores of XAI evaluation metrics corresponding to the XAI properties requested by the user. Like the precedent component, this one also acts as a wrapper that standardizes the input and output of XAI evaluation metrics and serves as a base to include any XAI evaluation metrics.

It also aggregates the properties' scores to provide a unique optimization objective for HPO. To find the best hyperparameters h of the XAI solution f_e, we define the optimization objective as follows:

$$\max_{h \in H} A(f_e, h) \tag{1}$$

With A the aggregation function and H the set of every possible hyperparameters. A should gather the multiple scores returned by the XAI evaluation metrics assessing the properties chosen and weighted according to the user's preferences for properties. To do so, we opt for a linear scalarization [32]:

$$A(f_e, h) = \frac{1}{c'} \sum_{q=1}^{c'} w_q \times sc_q(m_q(f_e^{(h)})) \tag{2}$$

The XAI evaluation metric for p_q is written $m_q(f_e^{(h)})$ for a shorter notation, thought it might use any partition of $\{X, F, F_e, Y\}$ as defined in 3.6. c' is the number of chosen properties and therefore XAI evaluation metrics. The weights w_q are the degree of importance set by the user and $sc_q(.)$ is a scaling function based on previous results for a property p_q. This scaling function allows not to favor one XAI evaluation metric over another. For the first epoch, there are no previous results and scaling cannot be defined. Therefore, we initialize $sc_q(.)$ using the evaluation scores of XAI solutions with the default hyperparameters. This cold start also verifies whether XAI solutions can perform well with default hyperparameters.

Although this framework already produces a ranking of XAI solutions, the computation time should be taken into account.

5 TIME-SAVING EVALUATION STRATEGIES

Some XAI solutions and XAI evaluation metrics were not designed to be used multiple times in a row and have high algorithmic complexity. To reduce the time cost of these algorithms, without changing their architecture, we propose to adapt the existing heuristic strategies from the AutoML field. AutoXAI adapts three AutoML strategies to reduce computing time: low fidelity which becomes sampling, early stopping, and weight sharing which becomes information sharing. These strategies are detailed below.

5.0.1 *Sampling.*

Reducing the number of explanations produced reduces the number of operations in explanation making and in explanation evaluation. It is also possible to use a subset of the dataset to create explanations, which can reduce the number of operations for explanation making. These approximations can be accurate enough depending on the diversity of the dataset, the complexity of the model, and the sensitivity of the XAI solution. This issue is addressed by specifying the percentage of explanations to be processed.

5.0.2 *Early stopping.*

Early stopping can be performed during the calculation of the XAI evaluation metric or the HPO. In both cases, this strategy can save a lot of time but it must be set up correctly to avoid approximating too roughly. For the calculation of evaluation metrics, this option is only possible if the XAI evaluation metric is applied sequentially to the explanations. Moreover, this strategy is more efficient if explanations are also calculated sequentially, indeed, fewer explanations are computed this way. The stopping condition is that the XAI evaluation metric only updates itself by a small percentage below a threshold during multiple iterations. We then consider that it stabilizes. For early stopping applied to HPO, the reasoning is that if the best score does not change during several iterations, then it has been found. Here, the choice of a threshold matters to avoid missing a better solution.

5.0.3 *Information sharing.*

Another way to save computing time is to reuse intermediate results and share information between evaluations. In AutoML, the weight-sharing strategy of an old model speeds up the training of a new one [19]. In AutoXAI, some intermediate calculations can be reused in the same way. This strategy is especially efficient for some costly intermediate calculations like a Gaussian process [29].

6 EXPERIMENTS

In the following experiments, an implementation of the AutoXAI framework described in Section 4 is applied to two use cases including the illustrative example described in Section 3.1. The code to reproduce the results of these use cases is available at https://github.com/RobinCugny/AutoXAI.

6.1 Diabetes estimation

For the illustrative example, the datasets used are diabetes dataset [30] and Pima Indians dataset [33]. Diabetes dataset has 10 features and is designed for a regression task to predict disease progression. Pima Indians dataset has 8 features and is made for a binary classification to predict if patients have diabetes. The black box model used is the scikit-learn implementation of a Multilayer Perceptron [34]. For the regression we use MLPRegressor[1] and for the classification we use MLPClassifier[2].

The implemented XAI solutions are LIME [5] and Kernel SHAP [18]. LIME provides explanations using the weights of a local linear model for each observation. SHAP captures features' interactions using Shapley values [35] as previously done by [36]. We use Kernel SHAP, one of their contributions, that builds a linear model like LIME but uses Shapley values as coefficients and therefore as feature importance. In this paper, SHAP refers to Kernel SHAP for short.

The implemented XAI evaluation metrics and their corresponding properties are:

- **Robustness** [29] *Continuity*
- **Infidelity** [27] *Correctness*
- **Number of features** [37] *Compactness*

[29] proposes the measure of **robustness** that evaluates *Continuity* by adapting Lipschitzian continuity. The objective is to measure changes in explanations while the input has small perturbations. Indeed, explanations should not radically change if the observation does not either. [27] proposes to evaluate *Correctness* with an **infidelity** metric. It consists in perturbing the input, based on the order of the features in explanations, and measuring for each new input the change in the output of the predictive function f. The *Compactness* is evaluated with the **number of features** which is obtained with the cardinal of the explanation vector. See Table 1 for the formulas.

For the aggregation in this scenario, Alice and Bob set the weights to 1, 2, 0.5 for **robustness**, **infidelity** and **number of features** respectively. Alice, sets the number of epochs to 25.

The HPO strategy is a Bayesian optimization, we use [31] implementation of the Gaussian Process. The time-saving evaluation strategies are early stopping for XAI evaluation metrics computation and HPO, and information sharing for robustness and infidelity. For robustness, information shared between epochs are the data points giving the maximum score of robustness (see Table 1) we call maxima for short. For infidelity, the information shared between epochs are the generated perturbation points and the model predictions for them.

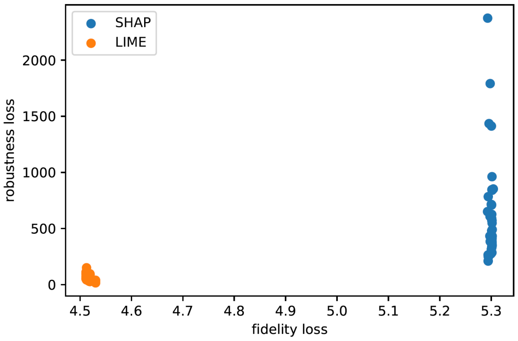


Figure 2: Robustness loss and Fidelity loss for diabetes dataset

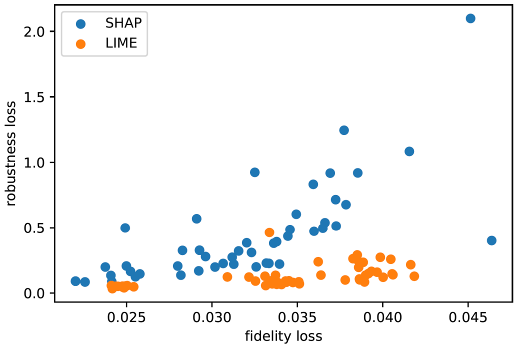


Figure 3: Robustness loss and Fidelity loss for Pima Indians dataset

An extract from the ranking produced by AutoXAI for Diabetes dataset is in Table 2 and the one for Pima Indians dataset is in Table 3. The XAI solutions are sorted in descending order according to the Aggregated score produced by Equation 2. To show diverse XAI solutions we present three combinations of hyperparameters with LIME and three with SHAP. The choice of the number of features in the explanation is a subjective process requiring to visualize the explanations, we decide on a number of features of 1, 3, and 5. Thus, the user can verify if short explanations are enough to understand the prediction or if more features would help. In the Hyperparameter columns, the two first hyperparameters for LIME as for SHAP are: first, the number of features in the explanation, and second, the number of perturbations used to build the linear model. The last hyperparameter for SHAP is the l1 regularization to use for feature selection[3].

Regarding Table 2, LIME is systematically higher than SHAP in the rankings with these XAI evaluation metrics. Several factors could explain why SHAP does not score better: with default hyperparameters, SHAP has lower average robustness than LIME on certain UCI classification datasets [38] (glass, wine, and leukemia)

[1] https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html

[2] https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

[3] https://shap-lrjball.readthedocs.io/en/latest/generated/shap.KernelExplainer.html

Table 1: XAI evaluation metrics

Property	Metric	Formula
Continuity	**Robustness**	$\hat{L}(x_i) = \max\limits_{x_j \in B_\epsilon(x_i)} \frac{\|f_e^{(h)}(x_i) - f_e^{(h)}(x_j)\|_2}{\|x_i - x_j\|_2}$ With $B_\epsilon(x_i)$ a ball of radius ϵ* centered in x_i the studied data point. The optimization method looks for the point in the data space with the highest ratio in the neighborhood. This value is kept as a measure for the lack of robustness $\hat{L}(x_i)$.
Correctness	**Infidelity**	$INFD(e_i, f, x_i) = \mathbb{E}_{I \sim \mu I}[(I^T e_i - (f(x_i) - f(x_i - I)))^2]$ With I, perturbations around x_i so that $I = x_i - x_i'$, here we choose a noisy implementation with $x_i' = x_i + \epsilon$ and ϵ a uniform noise.
Compactness	**Number of features**	$NoF(e_i) = \text{Card}(e_i)$ With $e_i \in E$, the i-th explanation vector, a feature summary.
Completeness	**Non-representativeness**	$NR(E) = \frac{1}{n} \sum\limits_{i=1}^{n} \min\limits_{e_j \in E} d(x_i, e_j)$ With $n = \text{Card}(X)$, d a distance function and E the explanations set composed of representative data points e_j also called prototypes.
Compactness *(Redundancy)*	**Diversity**	$Div(E) = \sum\limits_{\{e_i, e_j\} \in P_2(E)} \frac{d(e_i, e_j)}{C_2^l}$ With $P_2(E)$, the set of combinations of two prototypes, d a distance function and C_2^l the number of combination with $l = \text{Card}(E)$ prototypes.
Compactness *(Size)*	**Number of prototypes**	$NoP(E) = \text{Card}(E)$ With E, the prototypes set.

*The implementation is a box having the size of the standard deviation, it is a variation proposed by the original authors.

Table 2: Extract from the ranking produced by AutoXAI on Diabetes dataset.

Aggregated score	Scaled Robustness	Scaled Fidelity	Scaled NoF	XAI Solution	Hyperparameters
1.023	0.727	0.833	1.351	LIME	1;3656
1.019	0.703	0.991	0.745	LIME	3;8782
0.963	0.682	1.068	0.139	LIME	5;5392
-0.287	0.310	-0.924	1.351	SHAP	1;1304;auto
-0.633	-0.319	-0.975	0.745	SHAP	3;1571;aic
-0.639	0.014	-1.000	0.139	SHAP	5;1148;aic

Table 3: Extract from the ranking produced by AutoXAI on Pima Indians dataset.

Aggregated score	Scaled Robustness	Scaled Fidelity	Scaled NoF	XAI Solution	Hyperparameters
1.412	0.744	1.435	1.243	LIME	1;5347
1.282	0.575	1.325	1.243	SHAP	1;509;bic
0.361	0.633	0.117	0.430	LIME	3;8329
0.176	0.339	-0.014	0.430	SHAP	3;713;auto
0.070	0.262	0.070	-0.383	SHAP	5;537;bic
-0.185	0.599	-0.481	-0.383	LIME	5;7023

and especially a larger standard deviation [29]. For Diabetes dataset, as we can see in Figure 2, we also observe a large standard deviation and higher average robustness for SHAP with different hyperparameters. We observe especially that it has constantly a higher fidelity loss with a narrow distribution on this score. According to [27], this means that SHAP captures less well how the model prediction changes in response to the perturbations. As Kernel SHAP relies on LIME strategy to build a local linear model, this means that the loss might come from the inclusion of Shapley values. A hypothesis here would be that the black box model does not make much use of the relationships between features or not in the same way as Kernel SHAP detects it with its linear approximation. Besides, a similar conclusion has already been observed in [39].

Regarding the Pima Indians dataset, however, it seems that SHAP performs slightly better. In Figure 3, we can see that SHAP is less robust than LIME most of the time but that it has equivalent fidelity. Here SHAP seems to succeed in capturing the predictor function changes, therefore it might find more feature interactions in common with the model.

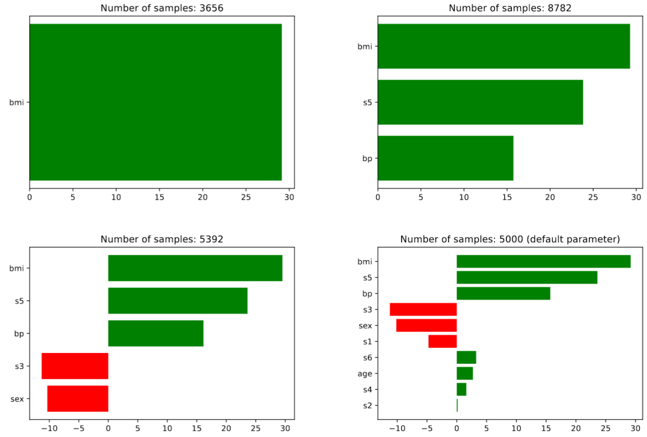


Figure 4: Different sizes of explanations produced by LIME for an observation from Diabetes dataset

Compactness has an impact on the other properties [15]. Moreover, Bob, the physician, should observe the explanations to confirm the number of features that are necessary to understand the prediction. Thus, it is appropriate to compare the XAI solutions using the aggregated score while taking into account the number of features. For that, let's pick a particular observation from the Diabetes dataset. The model makes a prediction and Bob asks *Why this prediction?* and wants to know the features contributing to this prediction. The explanations produced by the XAI solutions recommended by AutoXAI are in Figure 4. At the bottom right is LIME with default hyperparameters. We can see that some features have little influence on the prediction and are useless to answer the question of Bob. With these different-sized explanations, Bob can see what is important and what is negligible to him. He can thus choose the size of explanation he wants, keeping in mind the scores of the properties and the aggregated score.

6.2 SPAM detection

For the second use case, let's first describe the context. We consider a computer security manager in a laboratory, Charli. The staff complains about receiving spam in the chat service of the laboratory. Charli manages this service and knows little about ML, Charli wants to use an ML algorithm to automatically detect spam. Therefore, Charli finds an off-the-shelf GloVe embedding [40] and implement a LSTM [41]. Charli trains the model on a spam dataset, obtains a good accuracy, and decides to try it one week on the chat service. At the end of this week, a colleague of Charli shows a message that has not been sent because suspected as a spam. Therefore, Charli ends up wondering "Where does the model fail ?", more specifically, "What do false positives and false negatives look like?". To answer these questions, Charli wants to implement an XAI solution. In the end, Charli wants to use this knowledge to be able to explain to users the decisions made with their data and possibly reduce the error rate.

Charli needs examples of messages with specific predictions to know what they look like. Therefore, we define $\mathcal{E}_{i'}$ =*What kind of data lead to this prediction?* and $\mathcal{E}'_{j'}$ =*data points as explanations* (also called prototypes).

For the constraints of the context, Charli does not want to miss any type of message leading to an error. Ideally, Charli wishes that each data point should have a similar prototype (completeness). However, Charli also wants to avoid having too many prototypes and avoid redundancies (compactness).

The dataset used is derived from the UCI SMS Spam dataset [38], it has 8714 features for 5572 data samples and was built using TFIDF [42]. Charli separates the dataset into 4 subsets according to the model results: true positives, true negatives, false positives, and false negatives. Thus, the ML model (GloVe and LSTM) is no longer necessary for this experiment. The implemented XAI solutions are MMD-critic [43], Protodash [44] and k-medoids [45].

MMD-critic [43] proposes prototypes as explanations. To accurately represent the data distribution, it minimizes the discrepancy between the prototypes distribution and the data distribution. It also proposes critics which are points that are not well represented by the prototypes. Protodash [44] generalizes [43], it is a fast prototype selection that also associates non-negative weights to prototypes which are indicative of their importance. Eventually, although it was not proposed for XAI, we use k-medoids [45] as it is a baseline that gives comparable results. It finds medoids (prototypes here) such that the distance between one prototype and the other points of its data group is minimal.

The implemented XAI evaluation metrics and their corresponding properties are:

- **Non-representativeness** *Completeness*
- **Diversity** [13] *Compactness* (*redundancy*)
- **Number of prototypes** *Compactness* (*size*)

We propose a new **non-representativeness** metric to assess whether there is a close prototype for each data sample on average. Unlike [13], it is model-agnostic, as XAI solutions that do not use an ML model should not be evaluated according to the ML model. To assess the *Compactness*, we use the **number of prototypes** for the size of the explanation and **diversity**. For **diversity**, we adapt [13] proposal to measure the mean distance between prototypes. As

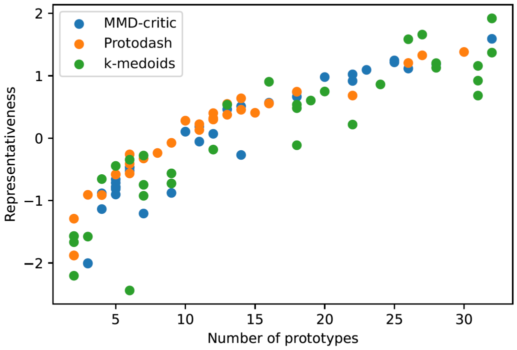


Figure 5: Influence of the number of prototypes on representativeness

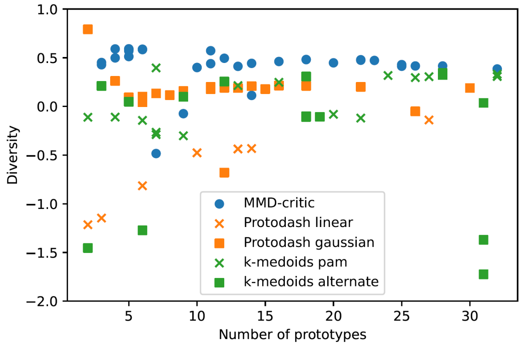


Figure 6: Influence of the number of prototypes on diversity

diversity and **number of prototypes** are fundamentally different, we consider that they correspond to two different sub-properties (*redundancy* and *size* respectively) and let the user give a weight to each. See Table 1 for the formulas.

For the aggregation in this scenario, Charli sets the weights to 2, 1, 2 for **non-representativeness**, **diversity** and **number of prototypes** respectively. Charli, sets the number of epochs to 25. The HPO strategy is also the Gaussian Process here.

An extract from the ranking produced by AutoXAI for SMS Spam dataset is in Table 4. As previously, the XAI solutions are sorted in descending order according to the Aggregated score. For each XAI solution, we present 2 results, the overall best score and the best score for a number of prototypes lower or equal to 5. In the Hyperparameter columns, k-medoids has the following hyperparameters: the initialization method, the maximum number of iterations, the algorithm to use, the metric and the number of medoids to generate [4]. MMD-critic has the following hyperparameters: the gamma value and the number of prototypes to find. Protodash has the following hyperparameters: the kernel to use, the value of sigma, and the number of prototypes to find.

Regarding Table 4, we observe that k-medoids has the best Aggregated score with high representativeness. We also observe that the representativeness score is systematically lower with fewer

[4] https://scikit-learn-extra.readthedocs.io/en/stable/generated/sklearn_extra.cluster.KMedoids.html

Table 4: Extract from the ranking produced by AutoXAI on SMS Spam dataset

Aggregated score	Scaled Representativeness	Scaled Diversity	Scaled NoP	XAI solution	Hyperparameters
0.483	0.904	0.248	-0.303	k-medoids	heuristic;300;pam;cosine;16
0.466	0.463	0.412	0.030	MMD-critic	1.000;13
0.384	0.224	0.201	0.251	Protodash	gaussian;11.90;11
0.367	-0.660	0.589	0.917	MMD-critic	1.000;5
0.331	-0.444	0.048	0.917	k-medoids	build;224;alternate;cosine;5
0.255	-0.580	0.092	0.917	Protodash	gaussian;21.43;5

Table 5: Computation time and scores for LIME evaluation without and with the time-saving evaluation strategies

	No strategy		Early stopping		Information sharing		Both strategies	
	Time	Score	Time	Score	Time	Score	Time	Score
Robustness	488.04 ± 14.79	-70.67 ± 0.64	17.45 ± 1.37	-67.41 ± 3.26	34.15 ± 1.70	-69.22 ± 2.24	1.22 ± 0.13	-67.62 ± 3.12
Infidelity	11.84± 0.27	-5.29 ± 0.06	0.46 ± 0.05	-5.00 ± 0.78	9.32 ± 0.37	-5.40 ± 0.12	0.35 ± 0.03	-5.48 ± 0.6

Computation times are in seconds. For evaluation scores, the higher, the better.

prototypes. This trend is shown in Figure 5. It can be seen that representativeness is more important with more prototypes. Indeed, intuitively, the more prototypes there are, the more likely it is that a data point is close to one of them. It results in a trade-off between compactness and completeness that encourage choosing appropriate weights for the properties. In Table 4, we observe that the two algorithms for k-medoids (pam and alternate) are performing well, while Protodash seems to have better results with the Gaussian kernel. This is confirmed in Figure 6 where Protodash performs better in terms of diversity with Gaussian kernel while k-medoids have equivalent results with pam and alternate algorithm. MMD-critic regularly has the best scores in diversity and k-medoids has a higher variance in scores on both diversity and representativeness.

Using the best scoring XAI solution on the SMS Spam dataset, Charli obtains the following prototypes for false positives:

- Hey pple...$700 or $900 for 5 nights...Excellent location wif breakfast hamper!!!
- Unlimited texts. Limited minutes.

As well as the following protoypes for false negatives:

- FROM 88066 LOST £12 HELP
- Money i have won wining number 946 wot do i do next

With these representative examples, Charli can explain to users what kind of messages the model may misclassify and can work on the data and the ML model while tracking predictions for these messages.

6.3 Time-saving evaluation strategies

Preliminary results for time-saving evaluation strategies (see Section 5) are obtained on the first use case (see Section 6.1) with diabetes dataset and MLPRegressor model. The XAI solution is LIME with default hyperparameters and the XAI evaluation metrics are robustness and infidelity. AutoXAI is run on a laptop with a 2.40 GHz octa-core CPU. Table 5 shows the average with its standard deviation of computation time and evaluation score. Early stopping saves 96.42% of the time for robustness and 96.13% for infidelity. Information sharing saves 93% of the time for robustness and 21.26% of the time for infidelity. For Information sharing with robustness, the maxima used for computing scores are obtained with LIME with other hyperparameters, hence the scores difference. With infidelity, the generated perturbation points and their corresponding predictions are obtained with another seed which explains the small difference in score compared to the no strategy baseline. Using both strategies saves 99.75% of the time for robustness and 97% for infidelity.

7 CONCLUSION AND PERSPECTIVES

In this paper, we propose AutoXAI, a framework that recommends the best XAI solutions according to the context of its user. AutoXAI automates the tedious task of selecting an XAI solution and its hyperparameters. It produces a ranking of solutions taking into account the preferences of the user. Although AutoXAI is system-centric like AutoML, here the user specifies the needs, the constraints and chooses the XAI solution in the ranking. It saves time and does not require the user to have deep knowledge of XAI. AutoXAI can also serve a researcher who wants to test an XAI proposal and monitor results. Indeed, as AutoXAI tests multiple XAI evaluation metrics on multiple XAI solutions and hyperparameters, potential relationships could be discovered. In this paper, we show there might be a trade-off between properties. Compactness, in particular, should be monitored and the user should decide by checking the explanations. The choice of one explanation over another one may raise ethical issues. Indeed, encouraging too much particular properties of explanations can lead to bias. For instance, reducing the number of prototypes can lead to missing the less represented data samples. Alternatively, a small number of features in an explanation can hide a bias in a model.

A short-term work could complement AutoXAI with new adaptations of AutoML methods. Longer perspectives should be to apply AutoXAI in a real-world setting and analyze users' feedback to assess its usefulness and give opportunities to improve it. Lastly, studying the influence of XAI properties on each other will be an important topic of study in the field of XAI solution evaluation.

REFERENCES

[1] Aniek F. Markus, Jan A. Kors, and Peter R. Rijnbeek. The role of explainability in creating trustworthy artificial intelligence for health care: A comprehensive survey of the terminology, design choices, and evaluation strategies. Journal of Biomedical Informatics, 113:103655, 2021.

[2] Andy Brown, Aaron Tuor, Brian Hutchinson, and Nicole Nichols. Recurrent neural network attention mechanisms for interpretable system log anomaly detection. In Proceedings of the First Workshop on Machine Learning for Computing Systems, MLCS'18, New York, NY, USA, 2018. Association for Computing Machinery.

[3] Sarah Tan, Rich Caruana, Giles Hooker, and Yin Lou. Distill-and-compare: Auditing black-box models using transparent model distillation. In Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES '18, page 303–310, New York, NY, USA, 2018. Association for Computing Machinery.

[4] Daniel Omeiza, Helena Webb, Marina Jirotka, and Lars Kunze. Explanations in autonomous driving: A survey. IEEE Transactions on Intelligent Transportation Systems, pages 1–21, 2021.

[5] 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, KDD '16, page 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery.

[6] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). IEEE access, 6:52138–52160, 2018.

[7] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine learning interpretability: A survey on methods and metrics. Electronics, 8(8), 2019.

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

[9] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information Fusion, 58:82–115, 2020.

[10] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608, 2017.

[11] Zachary C. Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. Queue, 16(3):31–57, jun 2018.

[12] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In 2018 IEEE 5th International Conference on data science and advanced analytics (DSAA), pages 80–89. IEEE, 2018.

[13] An-phi Nguyen and María Rodríguez Martínez. On quantitative aspects of model interpretability. arXiv preprint arXiv:2007.07584, 2020.

[14] Jianlong Zhou, Amir H. Gandomi, Fang Chen, and Andreas Holzinger. Evaluating the quality of machine learning explanations: A survey on methods and metrics. Electronics, 10(5), 2021.

[15] Meike Nauta, Jan Trienes, Shreyasi Pathak, Elisa Nguyen, Michelle Peters, Yasmin Schmitt, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. arXiv preprint arXiv:2201.08164, 2022.

[16] Tom Vermeire, Thibault Laugel, Xavier Renard, David Martens, and Marcin Detyniecki. How to choose an explainability method? towards a methodical implementation of xai in practice. In Machine Learning and Principles and Practice of Knowledge Discovery in Databases, pages 521–533, Cham, 2021. Springer International Publishing.

[17] Sebastian Palacio, Adriano Lucieri, Mohsin Munir, Sheraz Ahmed, Jörn Hees, and Andreas Dengel. Xai handbook: Towards a unified framework for explainable ai. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, pages 3766–3775, October 2021.

[18] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, page 4768–4777, Red Hook, NY, USA, 2017. Curran Associates Inc.

[19] Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. Knowledge-Based Systems, 212:106622, 2021.

[20] Q. Vera Liao, Daniel Gruen, and Sarah Miller. Questioning the AI: Informing Design Practices for Explainable AI User Experiences, page 1–15. Association for Computing Machinery, New York, NY, USA, 2020.

[21] James Overton. Scientific explanation and computation. In Thomas Roth-Berghofer, Nava Tintarev, and David B. Leake, editors, Explanation-aware Computing, Papers from the 2011 IJCAI Workshop, Barcelona, Spain, July 16-17, 2011, pages 41–50, 2011.

[22] Marko Robnik-Šikonja and Marko Bohanec. Perturbation-Based Explanations of Prediction Models, pages 159–175. Springer International Publishing, Cham, 2018.

[23] Gediminas Adomavicius, Bamshad Mobasher, Francesco Ricci, and Alexander Tuzhilin. Context-aware recommender systems. AI Magazine, 32(3):67–80, Oct. 2011.

[24] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018.

[25] Vijay Arya, Rachel K. E. Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Q. Vera Liao, Ronny Luss, Aleksandra Mojsilović, Sami Mourad, Pablo Pedemonte, Ramya Raghavendra, John Richards, Prasanna Sattigeri, Karthikeyan Shanmugam, Moninder Singh, Kush R. Varshney, Dennis Wei, and Yunfeng Zhang. One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques, 2019.

[26] Janis Klaise, Arnaud Van Looveren, Giovanni Vacanti, and Alexandru Coca. Alibi explain: Algorithms for explaining machine learning models. Journal of Machine Learning Research, 22(181):1–7, 2021.

[27] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. On the (in)fidelity and sensitivity of explanations. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019.

[28] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A Benchmark for Interpretability Methods in Deep Neural Networks. Curran Associates Inc., Red Hook, NY, USA, 2019.

[29] David Alvarez-Melis and Tommi S Jaakkola. On the robustness of interpretability methods. arXiv preprint arXiv:1806.08049, 2018.

[30] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. The Annals of statistics, 32(2):407–499, 2004.

[31] Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014–.

[32] Kaisa Miettinen. Nonlinear multiobjective optimization, volume 12. Springer Science & Business Media, 2012.

[33] Jack W Smith, James E Everhart, WC Dickson, William C Knowler, and Robert Scott Johannes. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In Proceedings of the annual symposium on computer application in medical care, page 261. American Medical Informatics Association, 1988.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

[35] Lloyd S Shapley. A value for n-person games. In Contributions to the Theory of Games, pages 307—-317, 1953.

[36] Erik Strumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. 11:1–18, mar 2010.

[37] Avi Rosenfeld. Better metrics for evaluating explainable artificial intelligence. In Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '21, page 45–50, Richland, SC, 2021. International Foundation for Autonomous Agents and Multiagent Systems.

[38] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[39] Emmanuel Doumard, Julien Aligon, Elodie Escriva, Jean-Baptiste Excoffier, Paul Monsarrat, and Chantal Soulé-Dupuy. A comparative study of additive local explanation methods based on feature influences. In Kostas Stefanidis and Lukasz Golab, editors, Proceedings of the 24th International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data (DOLAP), Edinburgh, UK, March 29, 2022, volume 3130 of CEUR Workshop Proceedings, pages 31–40. CEUR-WS.org, 2022.

[40] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, 2014.

[41] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. Neural Computation, 9(8):1735–1780, 11 1997.

[42] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. Journal of documentation, 1972.

[43] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 29. Curran Associates, Inc., 2016.

[44] Karthik S. Gurumoorthy, Amit Dhurandhar, Guillermo A. Cecchi, and Charu C. Aggarwal. Efficient data representation by selecting prototypes with importance weights. In Jianyong Wang, Kyuseok Shim, and Xindong Wu, editors, 2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019, pages 260–269. IEEE, 2019.

[45] Leonard Kaufman and Peter J Rousseeuw. Partitioning around medoids (program pam). Finding groups in data: an introduction to cluster analysis, 344:68–125, 1990.