

TREPAN Reloaded: A Knowledge-driven Approach to Explaining Black-box Models

Roberto Confalonieri¹ and Tillman Weyde² and Tarek R. Besold¹ and Fermín Moscoso del Prado Martín¹

Abstract. Explainability in Artificial Intelligence has been revived as a topic of active research by the need of conveying safety and trust to users in the ‘how’ and ‘why’ of automated decision-making. Whilst a plethora of approaches have been developed for post-hoc explainability, only a few focus on how to use domain knowledge, and how this influences the understandability of global explanations from the users’ perspective. In this paper, we show how ontologies help the understandability of global post-hoc explanations, presented in the form of symbolic models. In particular, we build on TREPAN, an algorithm that explains artificial neural networks by means of decision trees, and we extend it to include ontologies modeling domain knowledge in the process of generating explanations. We present the results of a user study that measures the understandability of decision trees using a syntactic complexity measure, and through time and accuracy of responses as well as reported user confidence and understandability. The user study considers domains where explanations are critical, namely, in finance and medicine. The results show that decision trees generated with our algorithm, taking into account domain knowledge, are more understandable than those generated by standard TREPAN without the use of ontologies.

1 INTRODUCTION

In recent years, explainability has been identified as a key factor for the adoption of AI systems in a wide range of contexts [6, 16, 12, 21, 29, 23]. The emergence of intelligent systems in self-driving cars, medical diagnosis, insurance and financial services among others has shown that when decisions are taken or suggested by automated systems it is essential for practical, social, and increasingly legal reasons that an explanation can be provided to users, developers or regulators. As a case in point, the European Union’s General Data Protection Regulation (GDPR) stipulates a right to “*meaningful information about the logic involved*”—commonly interpreted as a ‘right to an explanation’—for consumers affected by an automatic decision [24].³

The reasons for equipping intelligent systems with explanation capabilities are not limited to user rights and acceptance. Explainability is also needed for designers and developers to enhance system robustness and enable diagnostics to prevent bias, unfairness and discrimination [22], as well as to increase trust by all users in *why* and

how decisions are made. Against that backdrop, increasing efforts are directed towards studying and provisioning explainable intelligent systems, both in industry and academia, sparked by initiatives like the DARPA Explainable Artificial Intelligence Program (XAI), and carried by a growing number of scientific conferences and workshops dedicated to explainability.

While interest in XAI had subsided together with that in expert systems after the mid-1980s [5, 36], recent successes in machine learning technology have brought explainability back into the focus. This has led to a plethora of new approaches for local and global *post-hoc* explanations of black-box models [14], for both autonomous and human-in-the-loop systems, aiming to achieve explainability without sacrificing system performance. Only a few of these approaches, however, focus on how to integrate and use domain knowledge to drive the explanation process (e.g., [33, 27]) or to measure the understandability of explanations of black-box models (e.g., [30]). For that reason an important foundational aspect of explainable AI remains hitherto mostly unexplored: Can the integration of domain knowledge as, e.g., modeled by means of ontologies, help the understandability of interpretable machine learning models?

To tackle this research question, we propose a neural-symbolic learning approach based on TREPAN [8], an algorithm devised in order to explain trained artificial neural networks by means of decision trees, and we extend it to take into account ontologies in the explanation generation process. In particular, we modify the logic of the algorithm when choosing split nodes, to prefer features associated with more general concepts in a domain ontology. Having explanations bounded to structured knowledge, in the form of ontologies, conveys two advantages. First, it enriches explanations (or the elements therein) with semantic information, and facilitates effective knowledge transmission to users. Second, it supports the customisation of the levels of specificity and generality of explanations to specific user profiles [15].

In this paper, we focus on the first advantage, and on measuring the impact of the ontology on the perceived understandability of surrogate decision trees. To evaluate our approach, we designed and conducted an experiment to measure the understandability of decision trees in domains where explanations are critical, namely the financial and medical domain. Our study shows that decision trees generated by our modified TREPAN algorithm taking domain knowledge into account are more understandable than those generated without the use of domain knowledge. Crucially, this enhanced understandability of the resulting trees is achieved with little compromise on the accuracy with which the resulting trees replicate the behaviour of the original neural network model.

The remainder of the paper is organised as follows. After introducing TREPAN, and the notion of ontologies (Section 2), we present our

¹ Telefónica Innovación Alpha, email: {roberto.confalonieri, tarek.besold, fermin.moscoso}@telefonica.com

² Dept. of Computer Science, City, University of London, email: t.e.veyde@city.ac.uk

³ Regulation (EU) 2016/679 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) [2016] OJ L119/1.

Algorithm 1 $\text{Trepan}(\text{Oracle}, \text{Training}, \text{Features})$.

```
Priority queue  $Q \leftarrow \emptyset$ 
Tree  $T \leftarrow \emptyset$ 
use  $\text{Oracle}$  to label examples in  $\text{Training}$ 
enqueue root node into  $Q$ 
while  $nr\_internal\_nodes < size\_limit$  do
  pop node  $n$  from  $Q$ 
  generate  $examples$  for  $n$ 
  use  $\text{Features}$  to build set of candidate  $splits$ 
  use  $\text{Examples}$  and  $\text{Oracle}$  to determine  $Best\_split$ 
  add  $n$  to  $T$ 
  for element  $c \in Best\_split$  do
    add  $c$  as child of  $n$ 
    if  $c$  is not a leaf according to the  $\text{Oracle}$  then
      enqueue node  $c$  into  $Q$  with negative
      information gain as priority
    end if
  end for
end while
Return  $T$ 
```

revised version of the algorithm that takes into account ontologies in the decision tree extraction (Section 3). In Section 4, we propose how to measure understandability of decision trees from a technical and a user perspective. Section 5 reports and analyses the results of our experiment. After discussing our approach (Section 6), Section 7 situates our results in the context of related contributions in XAI. Finally, Section 8 concludes the paper and outlines possible lines of future work.

2 PRELIMINARIES

In this section, we present the main foundations of our approach, namely, the TREPAN algorithm and ontologies.

2.1 The TREPAN algorithm

TREPAN is a tree induction algorithm that recursively extracts decision trees from oracles, in particular from feed-forward neural networks [8]. The original motivation behind the development of TREPAN was to approximate a neural network by means of a symbolic structure that is more interpretable than a neural network classification model. This was in the context of a wider interest in knowledge extraction from neural networks (see [33, 9] for an overview).

The pseudo-code for TREPAN is shown in Algorithm 1. TREPAN differs from conventional inductive learning algorithms as it uses an *oracle* to classify examples during the learning process. It generates new examples by sampling from distributions over the given examples and constraints, so that the amount of training data used to select splitting tests and to label leaves does not decrease with the depth of the tree. It expands a tree in a best-first manner by means of a priority queue by entropy, that prioritises nodes that have greater potential for improvement. Further details of the algorithm can be found in [8].

TREPAN stops the tree extraction process using two criteria: all nodes do not need to be further expanded because their entropy is low (they contain almost exclusively instances of a single class), or a predefined limit of the tree size (the number of nodes) is reached. Whilst TREPAN was designed to explain neural networks as the oracle, it is a model-agnostic algorithm and can be used to explain any other classification model.

Entity \sqsubseteq \top	Person \sqsubseteq PhysicalObject
AbstractObject \sqsubseteq Entity	Loan \sqsubseteq AbstractObject
PhysicalObject \sqsubseteq Entity	Gender \sqsubseteq Quality
Quality \sqsubseteq Entity	Male \sqsubseteq Gender
LoanApplicant \sqsubseteq Person $\sqcap \exists hasApplied.Loan$	Female \sqsubseteq Gender
Domain($hasApplied$) = Person	Range($hasApplied$) = Loan

Figure 1: An ontology excerpt for the loan domain.

In this paper, our objective is to improve the understandability of the decision trees extracted by TREPAN. To this end, we extend the algorithm to take into account an *information content* measure, that is derived using ontologies, and computed using the idea of concept refinement, as detailed below. In order to evaluate the performance of both the original and extended TREPAN algorithms, we measure the accuracy and the fidelity of the resulting decision trees. *Accuracy* is defined as the percentage of test-set examples that are correctly classified. In contrast, *fidelity* is defined as the percentage of test-set examples on which the classification made by a tree agrees with that provided by its neural-network counterpart. Notice that the crucial measure for assessing the quality of the reconstructed tree is the fidelity, as this is the direct measure of how well the tree’s behaviour mimics the original neural network.

2.2 Ontologies

An ontology is a set of formulae in an appropriate logical language with the purpose of describing a particular domain of interest, such as finance or medicine. The precise logic used is not crucial for our approach as the techniques introduced here apply to a variety of logics. For the sake of clarity we use description logics (DLs) as well-known ontology languages. We briefly introduce the DL \mathcal{EL}_{\perp} , a DL allowing only conjunctions, existential restrictions, and the empty concept \perp . For full details, see [2, 1]. \mathcal{EL}_{\perp} is widely used in biomedical ontologies for describing large terminologies and it is the base of the OWL 2 EL profile.

Syntactically, \mathcal{EL}_{\perp} is based on two disjoint sets N_C and N_R of *concept names* and *role names*, respectively. The set of \mathcal{EL}_{\perp} *concepts* is generated by the grammar

$$C ::= A \mid C \sqcap C \mid \exists R.C,$$

where $A \in N_C$ and $R \in N_R$. A *TBox* is a finite set of general concept inclusions (GCIs) of the form $C \sqsubseteq D$ where C and D are concepts. It stores the terminological knowledge regarding the relationships between concepts. An *ABox* is a finite set of assertions $C(a)$ and $R(a, b)$, which express knowledge about objects in the knowledge domain. An *ontology* is composed by a TBox and an ABox. In this paper, we focus on the TBox only, thus we will use the terms ontology and TBox interchangeably.

The semantics of \mathcal{EL}_{\perp} is based on *interpretations* of the form $I = (\Delta^I, \cdot^I)$, where Δ^I is a non-empty *domain*, and \cdot^I is a function mapping every individual name to an element of Δ^I , each concept name to a subset of the domain, and each role name to a binary relation on the domain. I satisfies $C \sqsubseteq D$ iff $C^I \subseteq D^I$ and I satisfies an assertion $C(a)$ ($R(a, b)$) iff $a^I \in C^I$ ($(a^I, b^I) \in R^I$). The interpretation I is a *model* of the TBox \mathcal{T} if it satisfies all the GCIs and all the assertions in \mathcal{T} . \mathcal{T} is *consistent* if it has a model. Given two concepts C and D , C is *subsumed* by D w.r.t. \mathcal{T} ($C \sqsubseteq_{\mathcal{T}} D$) if $C^I \subseteq D^I$ for every model I of \mathcal{T} . We write $C \equiv_{\mathcal{T}} D$ when $C \sqsubseteq_{\mathcal{T}} D$ and $D \sqsubseteq_{\mathcal{T}} C$. C is *strictly subsumed* by D w.r.t. \mathcal{T} ($C \sqsubset_{\mathcal{T}} D$) if $C \sqsubseteq_{\mathcal{T}} D$ and $C \not\equiv_{\mathcal{T}} D$.

Figure 1 shows an ontology excerpt modeling concepts and relations relevant to the *loan* domain. The precise formalisation of the

domain is not crucial at this point; different formalisations may exist, with different levels of granularity. The ontology structures the domain knowledge from the most *general* concept (e.g., Entity) to more *specific* concepts (e.g., LoanApplicant, Female, etc.). The subsumption relation (\sqsubseteq) induces a partial order among the concepts that can be built from a TBox \mathcal{T} . For instance, the Quality concept is more general than the Gender concept, and it is more specific than the Entity concept.

We will capture the degree of generality (resp. specificity) of a concept in terms of an information content measure that is based on concept refinement. The measure is defined in detail in Section 3 and serves as the basis for the subsequent extension of the TREPAN algorithm.

2.3 Concept refinement

The idea behind concept refinement is to make a concept more general or more specific by means of refinement operators. Refinement operators are well-known in Inductive Logic Programming, where they are used to learn concepts from examples [35]. Refinement operators for description logics were introduced in [19], and further developed in [7, 34]. In this setting, two types of refinement operators exist: specialisation refinement operators and generalisation refinement operators. While the former construct specialisations of hypotheses, the latter construct generalisations.

In this paper we focus on specialisation operators. A specialisation operator takes a concept C as input and returns a set of descriptions that are more specific than C by taking an ontology into account. The proposal laid out in this paper can make use of any such operators (see e.g., [7, 34, 26]). When a specific refinement operator is needed, as in the examples and in the experiments, we use the following definition of specialisation operator based on the *downcover* set of a concept C .

Definition 2.1. Given a Tbox \mathcal{T} , and a concept description C , the specialisation operator $\rho_{\mathcal{T}}(C)$ is defined as follows:

$$\rho_{\mathcal{T}}(C) \subseteq \text{DownCov}_{\mathcal{T}}(C),$$

where $\text{DownCov}_{\mathcal{T}}(C)$ is the set of concepts that are more specific (or less general) than C :

$$\begin{aligned} \text{DownCov}_{\mathcal{T}}(C) := & \{D \in \text{sub}(\mathcal{T}) \mid D \sqsubseteq_{\mathcal{T}} C \text{ and} \\ & \nexists D' \in \text{sub}(\mathcal{T}) \text{ with } D \sqsubset_{\mathcal{T}} D' \sqsubset_{\mathcal{T}} C\}. \end{aligned}$$

In the above definition, $\text{sub}(\mathcal{T})$ denotes the union of all the subconcepts in the axioms in \mathcal{T} , plus $\{\top, \perp\}$. For any given axiom $C \sqsubseteq D$ in \mathcal{T} , the set of its subconcepts is $\text{sub}(C \sqsubseteq D) = \text{sub}(C) \cup \text{sub}(D)$; also, $\text{sub}(C(a)) = \text{sub}(C)$. Notice that $\text{sub}(\mathcal{T})$ is a finite set.

A concept C is specialised by any of its most general specialisations that belong to $\text{sub}(\mathcal{T})$. Every concept can be specialised into \perp in a finite number of steps.

Definition 2.2. The unbounded finite iteration of the refinement operator ρ is defined as:

$$\rho_{\mathcal{T}}^*(C) = \bigcup_{i \geq 0} \rho_{\mathcal{T}}^i(C).$$

where $\rho_{\mathcal{T}}^i(C)$ is inductively defined as:

$$\begin{aligned} \rho_{\mathcal{T}}^0(C) &= \{C\}, \\ \rho_{\mathcal{T}}^{j+1}(C) &= \rho_{\mathcal{T}}^j(C) \cup \bigcup_{C' \in \rho_{\mathcal{T}}^j(C)} \rho_{\mathcal{T}}(C'), j \geq 0. \end{aligned}$$

Thus $\rho_{\mathcal{T}}^*(C)$ is the set of subconcepts of C w.r.t. \mathcal{T} . We will denote this set by $\text{subConcept}(C)$. Since $\text{sub}(\mathcal{T})$ is a finite set, the operator $\rho_{\mathcal{T}}^*(C)$ is finite, and it terminates. For a detailed analysis of properties of refinement operators in DLs we refer to [19, 7].

Example 1. Let us consider the concepts Entity, and LoanApplicant defined in the ontology in Figure 1. Then: $\rho_{\mathcal{T}}(\text{Entity}) \subseteq \{\text{Entity}, \text{AbstractObject}, \text{PhysicalObject}, \text{Quality}\}$; $\rho_{\mathcal{T}}^*(\text{Entity}) \subseteq \text{sub}(\mathcal{T}) \setminus \{\top\}$; $\rho_{\mathcal{T}}(\text{LoanApplicant}) = \rho_{\mathcal{T}}^*(\text{LoanApplicant}) \subseteq \{\text{LoanApplicant}, \perp\}$.

3 TREPAN RELOADED

Our aim is to create decision trees that are more understandable for humans by determining which features are more understandable for a user, and assigning priority in the tree generation process according to increased understandability. Our hypothesis, which we will validate in this paper, is that features are more understandable if they are associated to more general concepts present in an ontology.

To measure the degree of semantic generality or specificity of a concept, we consider its *information content* [32] as typically adopted in computational linguistics [28]. There it quantifies the information provided by a concept when appearing in a context. Classical information theoretic approaches compute the information content of a concept as the inverse of its appearance probability in a corpus, so that infrequent terms are considered more informative than frequent ones.

In ontologies, the information content can be computed either extrinsically from the concept occurrences (e.g., [28]), or intrinsically, according to the number of subsumed concepts modeled in the ontology. Here, we adopt the latter approach. We use this degree of generality to prioritise features that are more general (thus presenting less information content), as our assumption is that the decision tree becomes more understandable when it uses more general concepts. From a cognitive perspective this appears reasonable, since more general concepts have been found to be easier to understand and learn [13], and we test this assumption empirically below.

Definition 3.1. Given an ontology \mathcal{T} , the information content of a feature X_i is defined as:

$$\text{IC}(X_i) := \begin{cases} 1 - \frac{\log(|\text{subConcepts}(X_i)|)}{\log(|\text{sub}(\mathcal{T})|)} & \text{if } X_i \in \text{sub}(\mathcal{T}) \\ 0 & \text{otherwise.} \end{cases}$$

where $\text{subConcepts}(X_i)$ is the set of specialisations for X_i , and $\text{sub}(\mathcal{T})$ is the set of subconcepts that can be built from the axioms in the TBox \mathcal{T} of the ontology (see Section 2.2).

It can readily be seen that the values of IC are smaller for features associated to more general concepts, and larger for those associated to more specific concepts instead.

Example 2. Let us consider the concepts Entity, and LoanApplicant defined in the ontology in Figure 1 and the refinements in Example 1. The cardinality of $\text{sub}(\mathcal{T})$ is 13. The cardinality of $\text{subConcepts}(\text{Entity})$ and $\text{subConcepts}(\text{LoanApplicant})$ is 12 and 2 respectively. Then: $\text{IC}(\text{Entity}) = 0.04$, and $\text{IC}(\text{LoanApplicant}) = 0.73$.

Having a way to compute the information content of a feature X_i , we now propose to update the information gain used by TREPAN to give preference to features with a lower information content.

Definition 3.2. The information gain given the information content IC of a feature X_i is defined as:

$$IG'(X_i, S|IC) := \begin{cases} (1 - IC(X_i))IG(X_i, S) & \text{if } 0 < IC(X_i) < 1 \\ 0 & \text{otherwise.} \end{cases}$$

where $IG(X_i, S)$ is the information gain as usually defined in the decision tree literature.

According to the above equation, IG' of a feature is decreased by a certain proportion that varies depending on its information content, and is set to 0 either when the feature is not present in the ontology or when its information content is maximal.

Our assumption that using features associated with more general concepts in the creation of split nodes can enhance the understandability of the tree, is based on users being more familiar with more general concepts rather than more specialised ones. To validate this hypothesis we ran a survey-based online study with human participants. Before proceeding to the details of the study and the results, as a prerequisite we introduce two measures for the understandability of a decision tree—an *objective*, syntax-based and a *subjective*, performance-based one—in the following section.

4 UNDERSTANDABILITY OF DECISION TREES

Understandability depends not only on the characteristics of the tree itself, but also on the cognitive load experienced by users in using the decision model to classify instances, and in understanding the features in the model itself. However, for practical processing, understandability of decision trees needs to be approximated by an objective measure. We compare here two characterisations of the understandability of decision trees, approaching the topic from these two different perspectives:

- Understandability based on the syntactic complexity of a decision tree.
- Understandability based on users' performances, reflecting the cognitive load in carrying out tasks using a decision tree.

On the one hand, it is desirable to provide a technical characterisation of understandability that can give a certain control over the process of generating explanations. For instance, in TREPAN, experts might want to stop the extraction of decision trees that do not overcome a given tree size limit, do have a stable accuracy/fidelity, but have an increasing syntactic complexity.

Previous work attempting to measure the understandability of symbolic decision models (e.g., [17]), and decision trees in particular [25], proposed syntactic complexity measures based on the tree structure. The syntactic complexity of a decision tree can be measured, for instance, by counting the number of internal nodes in the tree or leaves, the number of symbols used in the splits (relevant especially for *m-of-n* splits), or the number of branches that decision nodes have.

For the sake of simplicity, we focus on the combination of two syntactic measures: the number of leaves n in a decision tree, and the number of branches b on the paths from the root of the tree to all the leaves in the decision tree. Based on the results in [25], we define the *syntactic complexity* of a decision tree as:

$$U(n, b) := \alpha \frac{n}{k} + (1 - \alpha) \frac{b}{k^2}. \quad (1)$$

with $\alpha \in [0, 1]$ being a tuning factor that adjusts the weight of n and b , and $k = 5$ being the coefficient of the linear regression built using the results in [25].

On the other hand, the syntactic complexity of decision trees does not necessarily capture the ease with which actual people can use the resulting trees. A direct measure of user understandability is how accurately a user can employ a given decision tree to perform a decision. An often more precise measure of cognitive difficulty in mental processing is the reaction time (RT) or response latency [11]. RT is a standard measure used by cognitive psychologists and has even become a staple measure of complexity in the domain of design and user interfaces [37]. In the following section we describe an experiment measuring the cost of processing in terms of accuracy, and RT (among other variables) for different types of decision trees.

An additional factor that has to be taken into account is the tree size. It seems very likely that trees of different sizes, irrespective of any actual complexity, present more difficulties for human understanding that are not necessarily linearly related to the increase in tree size. Therefore, properly understanding the effects on actual understandability requires explicitly controlling the tree sizes. For our experiments, we define three categories of tree sizes based on the number of internal nodes: *small* (the number of internal nodes is between 0 and 10), *medium* (the number of internal nodes is between 11 and 20), and *large* (the number of internal nodes is between 21 and 30).

5 EXPERIMENTAL EVALUATION

5.1 Methods

Materials. We used datasets from two different domains to evaluate our approach: finance and medicine. We used the Cleveland Heart Disease Data Set from the UCI archive⁴, and a loan dataset from Kaggle⁵. For each of them, we developed an ontology defining the main concepts and relevant relations (the heart and loan ontology contained 29 classes, 66 logical axioms and 28 classes, 65 logical axioms respectively). To extract decision trees using the TREPAN and TREPAN Reloaded algorithm, we trained two artificial neural networks implemented in *pytorch*. The neural networks we use in our experiments have a single layer of hidden units. The number of hidden units used for each network is chosen using cross-validation on the network's training set, and we use a validation set to decide when to stop training networks. The accuracy of the trained neural networks was of 85.98% and 94.65% for the loan and heart dataset respectively. In total, for each of the neural networks, we constructed six decision trees, varying their size (measured in number of nodes; i.e., small, medium, large), and whether or not an ontology had been used in generating them. In this manner, we obtained a total of twelve distinct decision trees (2 domains \times 3 sizes \times 2 ontology presence values). Figure 2 shows two examples of distilled decision trees. The (avg.) fidelity of the extracted trees was of 92.73% (TREPAN) 92.63% (TREPAN Reloaded) and 89.23% (TREPAN) 88.17% (TREPAN Reloaded) for the loan and heart dataset respectively (see also Table 3). Notice that since the trees are post-hoc explanations of the artificial neural network, the fidelities of the distilled trees, rather than their accuracies, are the crucial measure.

⁴ <http://archive.ics.uci.edu/ml/datasets/Heart+Disease>

⁵ <https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset>

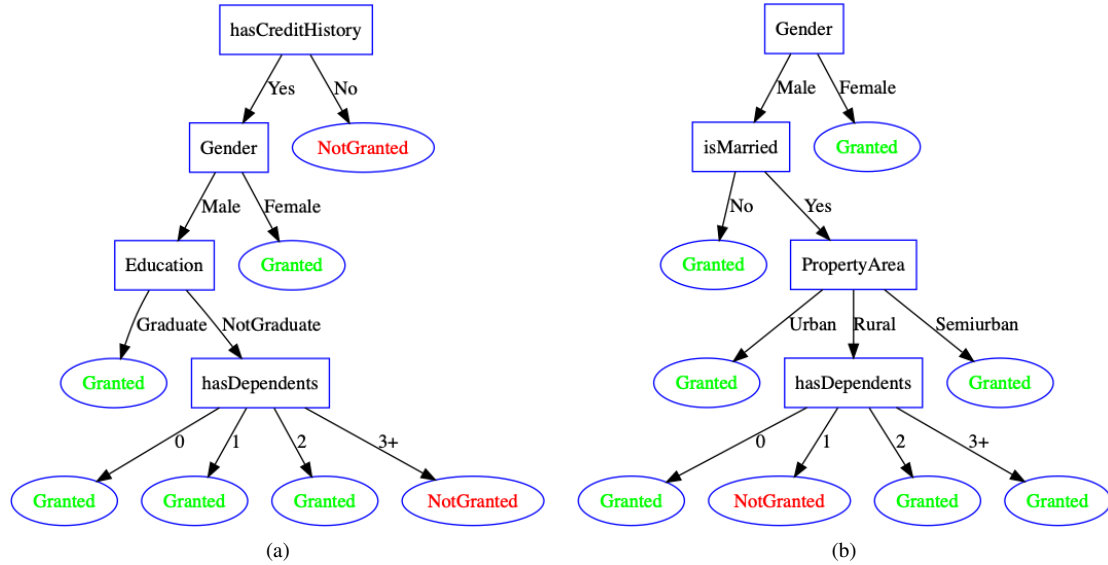


Figure 2: Decision trees of size ‘small’ in the loan domain, extracted without (left) and with (right) a domain ontology. As it can be seen the features used in the creation of the conditions in the split nodes are different.

Procedure. The experiment used two online questionnaires on the usage of decision trees. The questionnaires contained an introductory and an experimental phase.

In the introductory phase, subjects were shown a short video about decision trees, and how they are used for classification. In this phase, participants were asked to provide information on their age, gender, education, and on their familiarity with decision trees.

The experiment phase was subdivided into two tasks: classification, and inspection. Each task starts with an instruction page describing the task to be performed. In these tasks the participants were presented with the six trees corresponding to one of the two domains. In the classification task, subjects were asked to use a decision tree to assign one of two classes to a given case whose features are reported in a table (e.g., *Will the bank grant a loan to a male person, with 2 children, and a yearly income greater than €50,000,00?*). In the inspection task, participants had to decide on the truth value of a particular statement (e.g., *You are a male; your level of education affects your eligibility for a loan.*). The main difference between the two types of questions used in the two tasks is that the former provides all details necessary for performing the decision, whereas the latter only specifies whether a subset of the features influence the decision. In these two tasks, for each tree, we recorded:

- Correctness of the response.
- Confidence in the response, as provided on a scale from 1 to 5 (‘Totally not confident’=1, ..., ‘Very confident’=5).
- Response time measured from the moment the tree was presented.
- Perceived tree understandability as provided on a scale from 1 to 5 (‘Very difficult to understand’=1, ..., ‘Very easily understandable’=5).

Participants. 63 participants (46 females, 17 males) volunteered to take part in the experiment via an online survey.⁶ Of these 34 were exposed to trees from the finance domain, and 29 to those in the medical domain. The average age of the participants is 33 (\pm 12.23) years

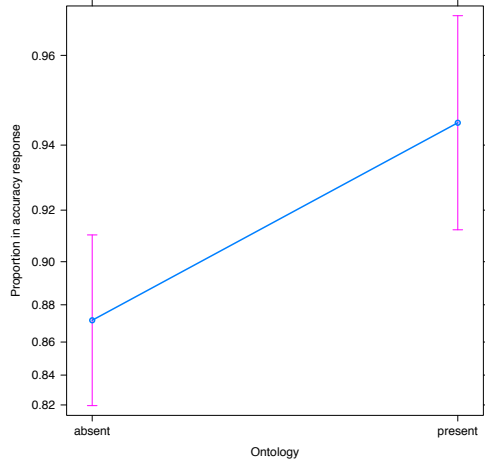
⁶ The participants were recruited among friends and acquaintances of the authors.

(range: 19 – 67). In terms of educational level their highest level was a Ph.D. for 28 of them, a Master degree for 9 of them, a Bachelor for 12, and a high school diploma for 14. 47 of the respondents reported some familiarity with the notion of decision trees, while 16 reported no such familiarity.

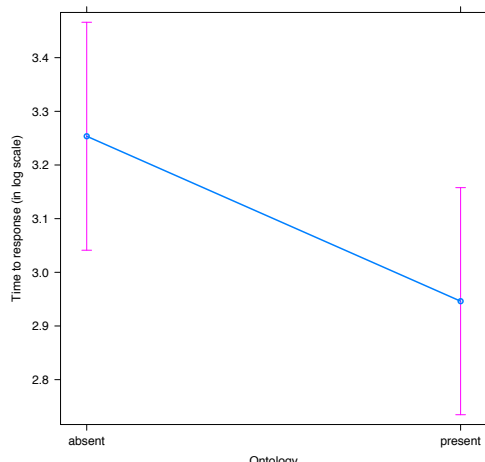
5.2 Results

We fitted a mixed-effects logistic regression model [3] predicting the correctness of the responses in the classification and inspection tasks. The independent fixed-effect predictors were the syntactic complexity of the tree, the presence or absence of an ontology in the tree generation, the task identity (classification vs. inspection), and the domain (financial vs. medical), as well as all possible interactions between them, as well as a random effect of the identity of the participant. A backwards elimination of factors revealed significant main effects of the task identity, indicating that responses were more accurate in the classification task than they were in the inspection ($z = -3.00, p = .0027$), of the syntactic complexity ($z = -3.47, p = .0005$), by which more complex tree produced less accurate responses, and of the presence of the ontology ($z = 3.70, p = .0002$), indicating that trees generated using the ontology indeed produced more accurate responses (Figure 3a). We did not observe any significant interactions or effect of the domain identity.

We analysed the response times (on the correct responses) using a linear mixed-effect regression model [3], with the log response time as the independent variable. As before, we included as possible fixed effects the task identity (classification vs. inspection), the domain (medical vs. financial), the syntactic complexity of the tree, and the presence or absence of ontology in the trees’ generation, as well as all possible interactions between them. In addition, we also included the identity of the participant as a random effect. A step-wise elimination of factors revealed main effects of task identity ($F(1, 593.87) = 20.81, p < .0001$), syntactic complexity ($F(1, 594.51) = 92.42, p < .0001$), ontology presence ($F(1, 594.95) = 51.75, p < .0001$), as well as signif-



(a)



(b)

Figure 3: Estimated main effects of ontology presence on accuracies (top) and time of response (bottom).

icant interactions between task identity and syntactic complexity ($F(1, 594.24) = 4.06, p = .0044$), and task identity and domain ($F(2, 107.48) = 5.03, p = .0008$). In line with what we observed in the accuracy analysis, we find that those trees that were generated using an ontology were processed faster than those that were generated without one (see Figure 3b).

We analysed the user confidence ratings using a linear mixed-effect regression model, with the confidence rating as the independent variable. We included as possible fixed effects the task identity (classification vs inspection), the domain (medical vs financial), the size of the tree, and the presence or absence of ontology in the trees' generation, as well as all possible interactions between them. In addition, we also included the identity of the participant as a random effect. A stepwise elimination of factors revealed a main effect of ontology presence ($F(1, 689) = 14.38, p = .0002$), as well as significant interactions between task identity and syntactic complexity ($F(2, 689) = 46.39, p < .0001$), and task identity and domain ($F(2, 110.67) = 3.11, p = .0484$). These results are almost identical to what was observed in the response time analysis: users show more confidence on judgments performed on trees that involved an ontology, the effect of syntactic complexity is most marked in the inspection task, and the difference between domains only affects the

classification task.

Finally, we also analysed the user rated understandability ratings using a linear mixed-effect regression model, with the confidence rating as the independent variable. We included as possible fixed effects the task identity (classification vs inspection), the domain (medical vs financial), the syntactic complexity of the tree, and the presence or absence of ontology in the trees' generation, as well as all possible interactions between them, and an additional random effect of the identity of the participant as a random effect. A stepwise elimination of factors revealed significant main effects of task ($F(1, 690) = 27.21, p < .0001$), syntactic complexity ($F(1, 690) = 104.67, p < .0001$), and of the presence of an ontology ($F(1, 690) = 39.90, p < .0001$). These results are in all relevant aspects almost identical to what was observed in the accuracy analysis: the inspection task is harder, more syntactically complex trees are less understandable than less complex ones, and trees originating from an ontology are perceived as more understandable.

6 DISCUSSION

Our hypothesis was that the use of ontologies to select features for conditions in split nodes, as described above, leads to decision trees that are easier to understand. This ease of understanding was measured theoretically using a syntactic complexity measure, and cognitively through time and accuracy of responses as well as reported user confidence and understandability.

First of all, the syntactic complexity (Eq. 1) of the trees distilled with TREPAN Reloaded is slightly smaller than those generated with TREPAN (see Table 1). Such small reduction on syntactic complexity might or might not reflect differences in the actual understandability of the distilled trees by people. However, in our experiments, all online implicit measures (accuracy and response time), and off-line explicit measures (user confidence and understandability ratings) indicate that trees generated using an ontology are significantly more accurately and easier understood by people than are trees generated without ontology. The analyses of the four measures are remarkably consistent in this crucial aspect (see Table 2).

Table 1: Syntactic complexity (Eq. 1) for trees inferred using C4.5, and distilled using TREPAN, and TREPAN Reloaded respectively.

	C4.5	TREPAN	TREPAN Reloaded
heart	5.64	3.56	3.46
loan	5.9	2.89	2.63

Table 2: Mean values of correct answers, time of response, user confidence, and user understandability for trees distilled using TREPAN and TREPAN Reloaded (standard deviations are reported in parenthesis). The difference in results is statistically significant w.r.t. Mann-Whitney and Wilcoxon tests for all measures.

Task	Measure	TREPAN	TREPAN Reloaded
Class.	%C. Answers	0.87 (0.32)	0.94 (0.18)
	Time (sec)	43.25 (61.16)	24.29 (15.67)
	Confidence	4.38 (0.86)	4.56 (0.80)
	Understd.	4.06 (0.97)	4.50 (0.48)
Insp.	%C. Answers	0.78 (0.41)	0.90 (0.27)
	Time (sec)	35.90 (24.80)	26.55 (35.74)
	Confidence	4.10 (0.96)	4.36 (0.78)
	Understd.	3.83 (1.01)	4.20 (0.87)

Table 3: Test-set accuracies and fidelities for trees distilled using TREPAN and TREPAN Reloaded.

	Accuracy				Fidelity	
	C4.5	NN	TREPAN	TREPAN Rld.	TREPAN	TREPAN Rld.
heart	81.97%	94.65%	82.43%	80.87%	89.23%	88.17%
loan	80.48%	85.98%	86.03%	82.80%	92.73%	92.63%

As we anticipated, coercing the outputs of TREPAN onto a pre-determined ontology (as in TREPAN reloaded) impacts the fidelity (and accuracy) of the resulting trees (see Table 3). Crucially, however, the very small compromise in the fidelity (on both examples, a drop of around one percent) of the neural network reconstruction is more than compensated for by the substantial improvement in the ease with which actual people can understand the resulting trees. When the goal is providing model explanations that are actually understandable by people, such a small compromise in fidelity is well worth it. Notice that, if we were not willing to compromise on fidelity at all, it would not make any sense to deviate in any amount from the original neural network’s performance (i.e., any fidelity below 100% would be unacceptable). In such case, however, one would retain the lack of user understandability of the models.

At this point, one might wonder why we should bother to create surrogate decision trees from black-box models, rather than inferring them directly from data. As already noticed in the original TREPAN work [8], distilling trees from networks can actually result in *better* trees than those one would obtain by building the decision trees directly. To demonstrate this point, we also trained decision trees directly from the datasets using the classical C4.5 algorithm. Table 3 shows that the trees inferred by the TREPAN variants are as accurate –if not more– than those inferred directly. Moreover, the trees built directly had syntactic complexities that roughly doubled those of the trees distilled using either TREPAN variant (see Table 1). This indicates that constructing trees directly from the data results in trees substantially more complex than those distilled by TREPAN variants, that nevertheless do not outperform them in the task.

There is a similarly small compromise in the accuracy of the decision trees (see Table 3). As we discussed above, in this approach, the accuracy of the resulting trees (i.e., their ability to replicate the testing sets) is less relevant than their fidelity (i.e., their ability to replicate the behaviour of the model we intend to explain). Nevertheless, our TREPAN Reloaded method improves the understandability of the trees w.r.t. the original TREPAN, while compromising little on the accuracy.

Apart from improving the understandability of (distilled) decision trees, ontologies also pave the way towards the capability of changing the level of abstraction of explanations to match different user profiles or audiences. For instance, the level of technicality used in an explanation for a medical doctor should not be the same as that used for lay users. One wants to adapt explanations without changing the underlying explanation procedure. Ontologies are amenable to automated abstractions to improve understandability [18]. The idea of concept refinement adopted here can be extended to operate on changing the definition of concepts and make them more general or more specific by means of refinement operators [7, 34, 26]. This is a line of work that we find a natural continuation of the current study.

In its current form, TREPAN Reloaded requires a predefined ontology onto which the features used by our algorithm should be mapped. In such cases, which are common in many domains (e.g., medical, pharmaceutical, legal, biological, etc.), one can directly apply TREPAN Reloaded to improve the quality of the explanations. Ad-

ditional work –beyond the scope of the current study– would be to automatically construct the most appropriate ontology to be mapped onto. Such a process could be achieved by automatically mapping sets of features into pre-existing general domain ontologies (e.g., MS Concept Graph [38], DBpedia [20]). The provision of some form of explicit knowledge, rather than being particular to our method, resides at the core of any attempts at human interpretable explanations. Whether such knowledge is in the form of a domain-specific ontology (as in this study), or as a domain-general one to be adapted ad-hoc, will depend on the particulars of specific applications.

7 RELATED WORKS

Most approaches on interpretable machine learning focus either on building directly interpretable models, or on reconstructing *post-hoc* local explanations. Our approach belongs to the category of post-hoc global explanation methods.

In this latter category, there are a few approaches that closely relate to ours. For instance, the work in [27] uses concepts to group features (either using expert knowledge or correlations), and embed them into surrogate models in order to constrain their training. In particular, the authors propose a revised version of TREPAN that considers only features belonging to concepts in the extraction of a decision tree. Whilst their results show that surrogate trees preserve accuracy and fidelity compared with original versions, the improvement in human-readability is not explicitly tested with users. The approach in [10] uses a complex neural network model to improve the accuracy of a simpler model, e.g., a simpler neural network, or a decision tree. This approach assumes to have a white-box access to (some of) the layers of the complex network model, whereas, in our approach we treat the black-box as an oracle. The authors in [4] describe a method for extracting decision tree explanations that actively samples new training points to avoid overfitting. Our approach is similar since TREPAN also uses new sampled data during the extraction of the decision tree.

Other works focus on building terminological decision trees from and using ontologies, e.g., [31, 39]). These approaches perform a classification task while building a tree rather than building a decision tree from a classification process computed by a black-box.

8 CONCLUSION AND FUTURE WORKS

In this paper, we proposed an extension of TREPAN, an algorithm that extracts global post-hoc explanations of black box-models in the form of decision trees. Our algorithm TREPAN Reloaded takes into account ontologies in the distillation of decision trees.

We showed that the use of ontologies ease the understanding of the distilled trees by actual users. We measured this ease of understanding through a rigorous experimental evaluation: theoretically, using a syntactic complexity measure, and, cognitively, through time and accuracy of responses as well as reported user confidence and understandability. All our measures indicated that trees distilled by TREPAN Reloaded are significantly more accurately and easier understood by people than are trees generated by TREPAN, with only little compromise of the accuracy and the fidelity (see Section 6).

The results obtained are very promising, and they open several direction of future research. On the one hand, we plan to extend this work to support the automatic generation of explanations that can accommodate different user profiles. On the other hand, we aim at investigating to apply our approach to explain CNNs in image classification (e.g., [40]). We also believe that this approach can be useful

in bias identification, to understand, for instance, if any undesirable discrimination features are affecting a black-box model.

REFERENCES

- [1] Franz Baader, Sebastian Brandt, and Carsten Lutz, ‘Pushing the \mathcal{EL} envelope.’, in *IJCAI*, pp. 364–369, (2005).
- [2] *The Description Logic Handbook: Theory, Implementation, and Applications*, eds., Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, Cambridge University Press, New York, NY, USA, 2003.
- [3] R.H. Baayen, D.J. Davidson, and D.M. Bates, ‘Mixed-effects modeling with crossed random effects for subjects and items’, *Journal of Memory and Language*, **59**(4), 390–412, (2008). Special Issue: Emerging Data Analysis.
- [4] Osbert Bastani, Carolyn Kim, and Hamsa Bastani, ‘Interpreting black-box models via model extraction’, *CoRR*, **abs/1705.08504**, (2017).
- [5] Bruce G. Buchanan and Edward H. Shortliffe, *Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley Longman Publishing Co., Inc., 1984.
- [6] Roberto Confalonieri, Tarek R. Besold, Tillman Weyde, Kathleen Creel, Tania Lombrozo, Shane Mueller, and Patrick Shafto, ‘What makes a good explanation? Cognitive dimensions of explaining intelligent machines’, in *Proc. of the 41th Annual Meeting of the Cognitive Science Society, CogSci 2019*, (2019).
- [7] Roberto Confalonieri, Manfred Eppe, Marco Schorlemmer, Oliver Kutz, Rafael Peñaloza, and Enric Plaza, ‘Upward refinement operators for conceptual blending in the description logic \mathcal{EL}^{++} ’, *Annals of Maths. and Artificial Intelligence*, **82**(1), 69–99, (2018).
- [8] Mark W. Craven and Jude W. Shavlik, ‘Extracting tree-structured representations of trained networks’, in *NIPS 1995*, pp. 24–30. MIT Press, (1995).
- [9] A. S. d’Avila Garcez, K. Broda, and D. M. Gabbay, ‘Symbolic knowledge extraction from trained neural networks: A sound approach’, *Art. Intell.*, **125**(1-2), 155–207, (2001).
- [10] Amit Dhurandhar, Karthikeyan Shanmugam, Ronny Luss, and Peder A Olsen, ‘Improving simple models with confidence profiles’, in *NIPS 2018*, 10296–10306, (2018).
- [11] Franciscus Cornelis Donders, ‘On the speed of mental processes.’, *Acta Psychologica*, **30**, 412–31, (1969).
- [12] Finale Doshi-Velez and Been Kim, ‘Towards a rigorous science of interpretable machine learning’, *CoRR*, **abs/1702.08608**, (2017).
- [13] Eleanor, Eleanor Rosch, Bascom Carolyn, Carolyn B Mervis, Wayne D. Gray, David M. Johnson, Jenifer L. Penny, and Penny Boyes-Braem, ‘Basic objects in natural categories’, *Cognitive Psychology*, **8**, 382–439, (1976).
- [14] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi, ‘A survey of methods for explaining black box models’, *ACM Comp. Surv.*, **51**(5), 1–42, (2018).
- [15] Michael Hind, ‘Explaining explainable ai’, *XRDS*, **25**(3), 16–19, (April 2019).
- [16] Robert R. Hoffman, Shane T. Mueller, Gary Klein, and Jordan Litman, ‘Metrics for explainable AI: challenges and prospects’, *CoRR*, **abs/1812.04608**, (2018).
- [17] Johan Huysmans, Karel Dejaeger, Christophe Mues, Jan Vanthienen, and Bart Baesens, ‘An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models’, *Decision Support Systems*, **51**(1), 141–154, (2011).
- [18] C. Maria Keet, ‘Enhancing Comprehension of Ontologies and Conceptual Models Through Abstractions’, in *AI*IA 2007: Artificial Intelligence and Human-Oriented Computing, 10th Congress of the Italian Association for Artificial Intelligence, Rome, Italy, September 10-13, 2007, Proceedings*, pp. 813–821, (2007).
- [19] Jens Lehmann and Pascal Hitzler, ‘Concept learning in description logics using refinement operators’, *Machine Learning*, **78**(1-2), 203–250, (2010).
- [20] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kon-tokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer, ‘DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia’, *Semantic Web Journal*, **6**(2), 167–195, (2015).
- [21] Zachary C. Lipton, ‘The mythos of model interpretability’, *Queue*, **16**(3), 30:31–30:57, (June 2018).
- [22] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan, ‘A survey on bias and fairness in machine learning, 2019.
- [23] Tim Miller, ‘Explanation in artificial intelligence: Insights from the social sciences’, *CoRR*, **abs/1706.07269**, (2017).
- [24] Parliament and Council of the European Union. General Data Protection Regulation, 2016.
- [25] Rok Piltaver, Mitja Luštrek, Matjaž Gams, and Sanda Martinčić-Ipšić, ‘What makes classification trees comprehensible?’, *Expert Syst. Appl.*, **62**(C), 333–346, (November 2016).
- [26] Daniele Porello, Nicolas Troquard, Rafael Peñaloza, Roberto Confalonieri, Pietro Galliani, and Oliver Kutz, ‘Two Approaches to Ontology Integration based on Axiom Weakening’, in *IJCAI-ECAI 2018*, pp. 1942–1948, (2018).
- [27] Xavier Renard, Nicolas Woloszko, Jonathan Aigrain, and Marcin Detynecki, ‘Concept tree: High-level representation of variables for more interpretable surrogate decision trees’, *CoRR*, **abs/1906.01297**, (2019).
- [28] Philip Resnik, ‘Using information content to evaluate semantic similarity in a taxonomy’, in *IJCAI 1995*, pp. 448–453. Morgan Kaufmann Publishers Inc., (1995).
- [29] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, ‘“Why Should I Trust You?”: Explaining the Predictions of Any Classifier’, in *Proc. of the 22nd Int. Conf. on Knowledge Discovery and Data Mining, KDD ’16*, pp. 1135–1144. ACM, (2016).
- [30] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin, ‘Anchors: High-precision model-agnostic explanations’, in *AAAI*, pp. 1527–1535. AAAI Press, (2018).
- [31] Giuseppe Rizzo, Claudia dAmato, Nicola Fanizzi, and Floriana Esposito, ‘Tree-based models for inductive classification on the web of data’, *Journal of Web Semantics*, **45**, 1–22, (2017).
- [32] David Sánchez, Montserrat Batet, and David Isern, ‘Ontology-based information content computation’, *Knowledge-Based Systems*, **24**(2), 297–303, (2011).
- [33] Geoffrey G. Towell and Jude W. Shavlik, ‘Extracting refined rules from knowledge-based neural networks’, *Machine Learning*, **13**(1), 71–101, (October 1993).
- [34] Nicolas Troquard, Roberto Confalonieri, Pietro Galliani, Rafael Peñaloza, Daniele Porello, and Oliver Kutz, ‘Repairing Ontologies via Axiom Weakening’, in *AAAI 2018*, pp. 1981–1988, (2018).
- [35] Patrick R.J. van der Laag and Shan-Hwei Nienhuys-Cheng, ‘Completeness and properness of refinement operators in inductive logic programming’, *The Journal of Logic Programming*, **34**(3), 201–225, (1998).
- [36] Michael R. Wick and William B. Thompson, ‘Reconstructive expert system explanation’, *Art. Intelligence*, **54**(1-2), 33–70, (March 1992).
- [37] Jill Butler William Lidwell, Kritina Holden, *Universal. Principles of Design.*, Rockport, 2003.
- [38] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q. Zhu, ‘Probase: A Probabilistic Taxonomy for Text Understanding’, in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, SIGMOD ’12*, pp. 481–492, New York, NY, USA, (2012). ACM.
- [39] Jun Zhang, Adrian Silvescu, and Vasant G. Honavar, ‘Ontology-driven induction of decision trees at multiple levels of abstraction’, in *Proc. of the 5th Int. Symposium on Abstraction, Reformulation and Approximation (SARA) 2002*, volume 2371 of *LNC3*, pp. 316–323. Springer, (2002).
- [40] Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu, ‘Interpreting cnns via decision trees’, in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (June 2019).