An Ontology-based Approach to Explaining Artificial Neural Networks

Roberto Confalonieri[1], Fermín Moscoso del Prado[1], Sebastia Agramunt[1], Daniel Malagarriga[1], Daniele Faggion[1], Tillman Weyde[2], and Tarek R. Besold[1]

[1] Alpha Health AI Lab @ Telefónica Innovación Alpha
Plaça d'Ernest Lluch i Martin, 5
E-08019 Barcelona
name.surname@telefonica.com
[2] City, University of London
Research Centre for Machine Learning,
Northampton Square, London EC1V 0HB
t.e.weyde@city.ac.uk

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 an explanation from the users' perspective. In this paper we show how ontologies help the understandability of interpretable machine learning models, such as decision trees. 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 in domains where explanations are critical, namely, in finance and medicine. Our study shows that decision trees taking into account domain knowledge during generation are more understandable than those generated without the use of ontologies.

Keywords: Explainable AI · Ontologies · Neural-Symbolic Learning

1 Introduction

In recent years, explainability has been identified as a potential key factor for the adoption of AI systems in a wide range of contexts [16,8,22,30,23]. 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 [27].[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, 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 [5,37], recent successes in machine learning technology have brought explainability back into the focus. This has led to a plethora of new approaches for *post-hoc* explanations of black-box models [13], for both autonomous and human-in-the-loop systems, aiming to achieve explainability without sacrificing system performance (see related works in Section 6). Only a few of these approaches, however, focus on how to integrate and use domain knowledge to drive the explanation process (e.g., [35]), or to measure the understandability of global and local explanations of black-box models (e.g., [31]). 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 [7], 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 in choosing split nodes, to prefer features associated to more abstract and general concepts in a domain ontology.

To evaluate our approach we designed and an experiment aiming at measuring 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 taking domain knowledge into account are more understandable than those generated without the use of domain knowledge.

The remainder of the paper is organised as follows. After introducing Trepan, and the notion of ontologies (Section 2), we present our 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 the results of our experimental evaluation. Section 6 gives an overview of related contributions in the context of XAI, before Section 7 considers possible lines of future work.

[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 Trepan($Oracle$,$Training$,$Features$)

Priority queue $Q \leftarrow \emptyset$
Tree $T \leftarrow \emptyset$
use $Oracle$ to label examples in $Training$
enqueue root node into Q
while $nr\_internal\_nodes < size\_limit$ **do**
    pop node n from Q
    generate *examples* for n
    use $Features$ to build set of candidate *splits*
    use *examples* and $Oracle$ to evaluate *best_split* from *splits*
    add n to T
    for split $s \in best\_splits$ **do**
        create a new node c as child of n
        if c is not a leaf according to the $Oracle$ **then**
            enqueue node c into Q
        end if
    end for
end while
Return T

2 Preliminaries

In this section, we present the main conceptual cornerstones 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 [7]. The original motivation behind the development of TREPAN was to approximate neural networks 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 [35,11] 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, and to decide when a node becomes a leaf node. Also, it generates new examples by sampling 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, that stores nodes that have greater potential to increase the *fidelity* of the tree to the oracle (the fraction of instances for which the tree and the oracle agree in their prediction). Further details on the algorithm can be found in [7].

TREPAN stops the tree extraction process using two criteria: all leaf nodes does not need to be further expanded (they contain almost exclusively instances of a single class), or a predefined limit of the tree size (the number of internal

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

Fig. 1: An ontology excerpt for the loan domain.

nodes) is reached. Whilst Trepan was designed to explain neural networks (the oracle is typically a neural network), it is a model-agnostic algorithm and can be used to explain any other black-box model.

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 refinements, as detailed below.

2.2 Ontologies

An ontology is a set of formulas in an appropriate logical language with the purpose of describing a particular domain of interest, for instance, finance or medicine. The precise logic used is not crucial for our approach as most techniques introduced apply to a variety of logics; however, 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 [3,2]. $\mathcal{EL}_\perp$ 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.

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 $\mathcal{I}$ is a *model* of the ontology $\mathcal{T}$ 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. the ontology $\mathcal{T}$ ($C \sqsubseteq_{\mathcal{T}} D$) 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 $C \not\equiv_{\mathcal{T}} D$.

We denote by $\mathsf{sub}(\mathcal{T})$ the set of *subconcepts* that can be built from $\mathcal{T}$. The set of *subconcepts* of $\mathcal{T}$ is given by $\{\top, \bot\} \cup \bigcup_{C \sqsubseteq D \in \mathcal{T}} \mathsf{sub}(C) \cup \mathsf{sub}(D)$ where $\mathsf{sub}(C)$ is defined by structural induction on C.

Figure 1 shows an excerpt of an ontology 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.[4] The ontology consists of axioms that structure the described domain knowledge from the most *general* concept (e.g., Entity) to more *specific* concepts (e.g., $\mathsf{LoanApplicant}$, Female, etc.). The subsumption relation ($\sqsubseteq$) induces a partial order among the concepts that can be built from a TBox $\mathcal{T}$ (in particular $\langle \mathsf{sub}(\mathcal{T}), \sqsubseteq_{\mathcal{T}} \rangle$ is a quasi-ordered set). For instance, the $\mathsf{Quality}$ 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.

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. 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 [19]. Refinement operators for description logics were introduced in [21].

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 $\mathcal{T}$ into account.

The proposal laid out in this paper can make use of any such operators (see e.g., [6,36]). 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.

$$\rho_{\mathcal{T}}(C) \subseteq \mathsf{DownCov}_{\mathcal{T}}(C).$$

[4] A discussion of the impact of ontology design and engineering can be found in [24]. Note that, even in an expressive language L and given a rich axiomatisation O of a concept C, the ontology O is thought to only approximately describe the intended models of C [12].

where $\mathsf{DownCov}_{\mathcal{T}}(C)$ is defined as:

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

That is, a concept C is specialised by any of its most general specialisations that belong to the set of subconcepts of $\mathcal{T}$. Notice $\mathsf{sub}(\mathcal{T})$ is a finite set, this guarantees the operator defined above to be finite. For an analysis of properties of refinement operators in DLs we refer to [6,21].

Then the unbounded finite iteration of the refinement operator ρ is defined as $\rho^*_{\mathcal{T}}(C) = \bigcup_{i\geq 0} \rho^i_{\mathcal{T}}(C)$ where $\rho^i_{\mathcal{T}}(C)$ is inductively defined as $\rho^0_{\mathcal{T}}(C) = \{C\}$, $\rho^{j+1}_{\mathcal{T}}(C) = \rho^j_{\mathcal{T}}(C) \cup \bigcup_{C' \in \rho^j_{\mathcal{T}}(C)} \rho_{\mathcal{T}}(C')$, $j \geq 0$.

Notice that every concept can be specialised into $\perp$ in a finite number of steps. Thus $\rho^*_{\mathcal{T}}(C)$ is the set of subconcepts of C wrt $\mathcal{T}$. We will denote this set by $\mathsf{subConcept}(C)$.

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

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.

Which specific properties make a concept more or less understandable is still an open question in cognitive science [20]. In this paper we build on the intuition that relating features to concepts belonging to a domain ontology—effectively tying the features into the domain context—increases understandability.

Given an ontology, we assume that features are more understandable if they are associated to more general concepts present in the ontology. To measure the degree of semantic generality or specificity of a concept, we consider its *information content* [34] as typically adopted in computational linguistics [29]. There it is used to quantify 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., [29]), 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. From a cognitive perspective this appears reasonable, since more general concepts have been found to be less difficult to understand and learn [9]. Therefore, the decision tree should also become more understandable.

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

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

where $\mathsf{subConcepts}(X_i)$ is the set of specialisations for X_i, and $\mathsf{sub}(\mathcal{T})$ is the set of subconcepts in the ontology $\mathcal{T}$ (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 $\mathsf{sub}(\mathcal{T})$ is 13. The cardinality of subConcepts(Entity) and subConcepts(LoanApplicant) is 12 and 2 respectively. Then: $\mathsf{IC}(\mathsf{Entity}) = 0.04$,and $\mathsf{IC}(\mathsf{LoanApplicant}) = 0.73$.

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 informational content. Thus, we define:

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

where $\mathsf{IG}(X_i, S)$ is the information gain as usually defined in the decision tree literature. According to the above equation, the information gain IG' of a feature is decreased by a certain quantity that varies depending on its informational content, and is set to 0 either when the feature is not present in the ontology or when its information content is maximal.

We assume that using features associated with more general concepts (thus presenting less information content) in the creation of split nodes can enhance the understandability of the tree, since users are 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 presentation of the study and the results in Section 5, 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

Measuring the *understandability* of decision trees is an ambitious task. 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.

In this paper, we compare two characterisations of the understandability of decision trees, approaching the topic from two different perspectives:

– Understandability based on the syntactic complexity of a decision tree.

– Understandability based on user's 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 allows 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 [28], 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 observations in [28], we characterise the *syntactic complexity* of a decision tree as

$$\alpha \frac{n}{k^2} + (1-\alpha)\frac{b}{k}.$$

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 [28].

Used as understandability metric, this provides a way to define the trade-off between fidelity and understandability in the extraction of decision trees, and thus to restrict the syntactical complexity of the trees extracted by TREPAN.

On the other hand, the syntactic complexity of decision trees is not sufficient to capture the cognitive load that users experience while using it to make decisions, and other metrics are needed. For this, in the following section we describe an experiment measuring different aspects of the cognitive cost of processing different types of decision trees.

An additional factor that has to be taken into account is the tree size as it can affect the usage of the trees. 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 heart dataset (Cleveland) from the UCI archive[5], and a loan dataset from Kaggle[6]. The heart dataset consists of 14

[5] `http://archive.ics.uci.edu/ml/datasets/Heart+Disease`
[6] `https://www.kaggle.com/hafidhfikri/loan-approval-prediction/notebook`

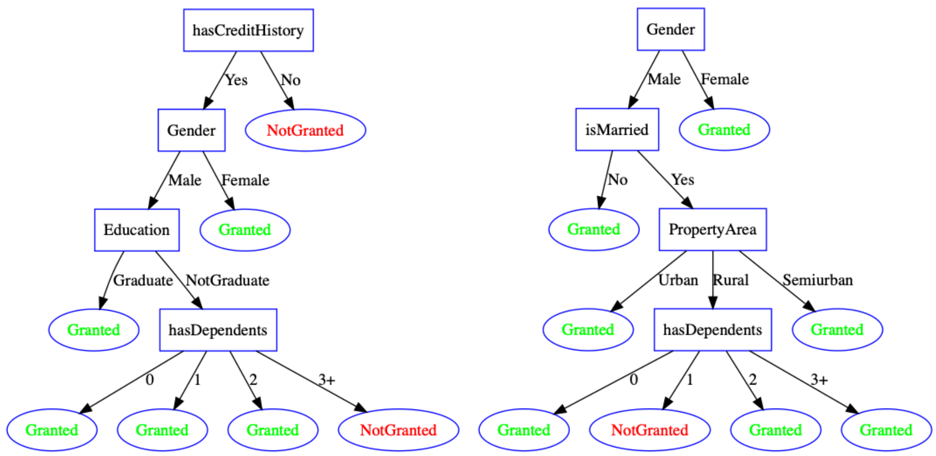

Fig. 2: Decision trees of size 'small' in the loan domain, extracted without (left) and with (right) a domain ontology. As it can be seen the features used in the creation of the conditions in the split nodes are different.

features and 303 instances. The loan dataset consists of 13 features and 614 instances. For each of them, we developed an ontology defining the main concepts and relevant relations. To extract decision trees using the Trepan and Trepan reloaded algorithm, we trained two artificial neural networks implemented in *pytorch*.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 extracted decision trees.

Procedure. The experiment used two 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 on the response, as provided on a scale from 1 to 5 ('Very Confident'=5, ..., 'Totally not Confident'=1).
- Response time measured from the moment the tree was presented.
- Perceived tree understandability as provided on a scale from 1 to 5 ('Very Easily Understandable'=5, ..., 'Very Difficult to Understand'=1).

Participants. 63 participants (46 females, 17 males) volunteered to take part in the experiment via an online survey. 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 (± 12.23) years (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 and Discussion

Our hypothesis is that the use of ontologies to select features for conditions in split nodes, as described above in Section 3, leads to decision trees that are easier to understand. This ease of understanding is measures through speed and accuracy of responses as well as reported confidence and understandability.

We fitted a mixed-effects logistic regression model [4] predicting the correctness of the responses in the classification and inspection tasks. The independent fixed-effect predictors were the size of the decision tree (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 ($z = -3.00, p = .0027$), 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. We did not observe any significant interactions or effect of the domain identity. The estimated effect sizes and directions are illustrated on Figure 3.

We analysed the response times (on the correct responses) using a linear mixed-effect regression model [4], with the log response time as the independent

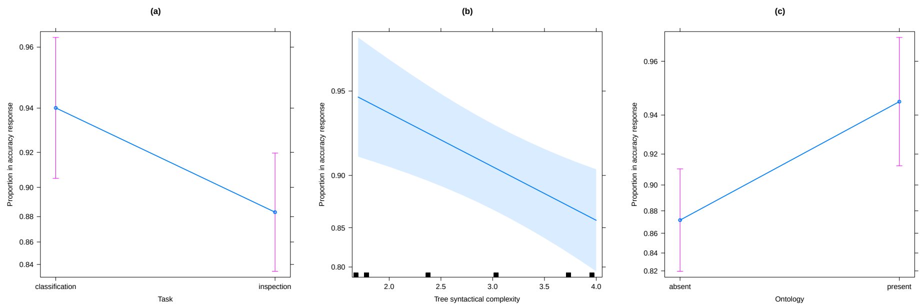

Fig. 3: Estimated main effects of task presence (a), syntactic complexity (b), and ontology presence (c) on accuracies.

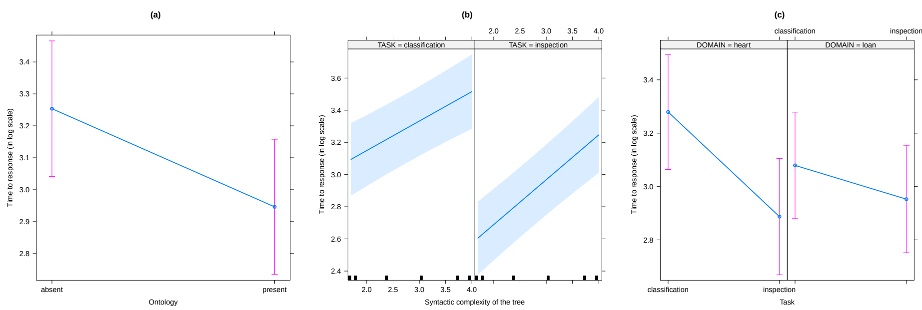

Fig. 4: Estimated main effects of ontology presence (a), and two-way interactions between task and syntactic complexity (b) and task and domain (c) on response time.

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 then. In addition, we also included the identity of the participant as a random effect. A stepwise 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 significant interactions between task identity and syntactic complexity ($F(1, 594.24) = 4.06, p = .0044$), and task identity and domain ($F(2, 107.48) = 5.03, p = .0008$).

Figure 4b plots the estimated interaction between syntactic complexity and task identity on the response times. Overall, across both cases the more complex trees result in longer response times (as was evidenced by the main effect of syntactic complexity). However, the interaction indicates that this effect is significantly more marked in the inspection task than it is in the classification task. This is in line with our intuition that the inspection task requires a more detailed examination of the decision tree, and it is therefore more sensitive to its complexity. Similarly, Figure 4c, depicts how the interaction between task and domain indicates that the response time difference between domains is only observable in the classification task (notice that there was no significant main effect of domain, just the interaction). Finally, and most crucially, 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 4a).

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 syntactic complexity of the tree, and the presence or absence of ontology in the trees' generation, as well as all possible interactions between then. 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 then, 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.

In summary, 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 more accurately and easier understood by people than are trees generated without them. The analyses of the four measures are remarkably consistent in this crucial aspect.

We also investigated whether using ontologies damaged the fidelity of the decision trees to the original neural network (Figure 5).[7] Trees generated using

[7] The accuracy of classification of the trained neural networks was of 85.98% and 94.65% for the loan and heart dataset respectively.

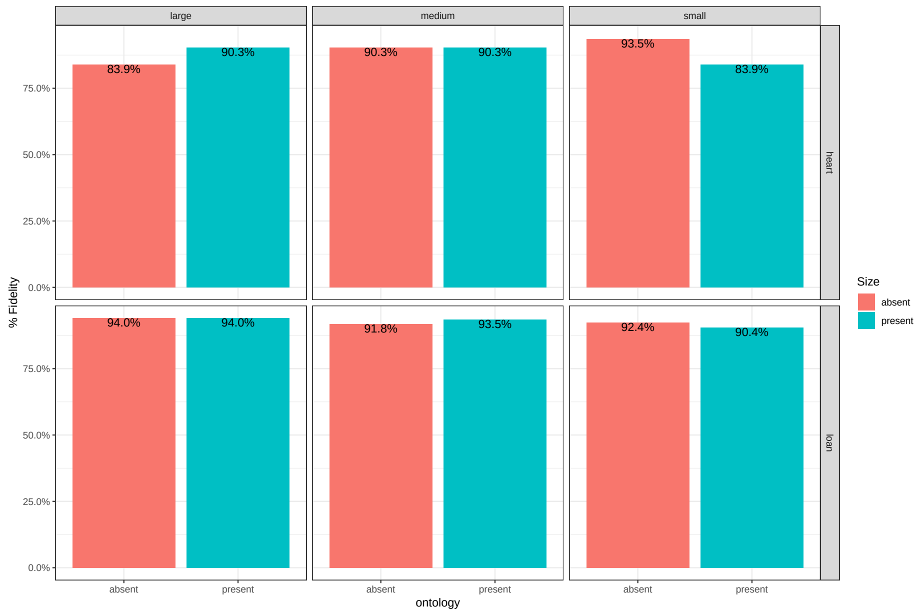

Fig. 5: Impact of the use of ontology on the fidelity of trees of different sizes (small, medium, large), and in the two domains considered (heart, loan). Percentages show that the fidelity of the tree is not much affected by the use of ontologies.

ontologies are slightly less accurate reconstructions of the neural network (91.0% vs 90.4% fidelity). However, this loss may be outweighed by the substantial gain in understandability, depending on the application.

6 Related Work

There exists some work focusing on building terminological decision trees from and using ontologies (e.g., [32,38]). These approaches focus on performing a classification task while building a tree rather than building a decision tree from a classification process computed by other method. Still they demonstrate how the use of ontologies can benefit non-symbolic tasks and i deserves to be explored.

On the other hand, the literature about making machine learning model interpretable and about extracting explanations, so that their decision behavior can be understood, is vast. A survey on interpretable machine learning methods and techniques can be found in [13]. Most of existing approaches for Explainable AI focus on reconstructing interpretable *post-hoc* approximations of black-box models. These approaches can be classified according to the type of model extracted: global, local, and introspective.

Approaches that build explanations as a global model typically extract decision trees [7,10] or decision rules [40,25] from black-box models, using the black-box as an oracle. The approach in this paper belong to this category. In some cases the interpretable model is a refinement of previous models, which were used to build the black box, e.g., [35].

Local model-based approaches build explanations for specific outcomes and instances of a black-box model. In this respect, explanations are considered as local approximations of how a black-box model works [31,30,18].

Finally, approaches based on introspective models build explanations relating inputs to outputs of a black-box model. For instance, explanations can consist of saliency masks for Deep Neural Networks in image classification [33,26,15], and groups of input-output tokens that are causally related [1].

More recently, there are efforts to design intelligent systems to be interpretable by design, e.g., in recommender systems [39], or in initiatives developing the concept of *perspicuous computing*[8].

7 Conclusion and Future Works

We showed how ontologies can be integrated in Trepan to take the degree of generality or specificity of features into account in the process of building a decision tree. We modified the information gain function of the induction learning algorithm by taking into account the generality in such a way that features associated to more general concept in a domain ontology are prioritised in the creation of the conditions in split nodes.

We evaluated our approach through a user study in which users were asked to carry out two decision tasks using trees generated without and with the use of ontologies. From the evaluation, we were able to conclude that the ontology plays a positive and significant role not only in the performance of the tasks, but also in the perceived understandability of the trees. The results obtained are thus very promising, and they open several direction of future research into explainability and the task of neural-symbolic integration, which is very challenging.

First, we plan to use this approach to study and prevent the creation of biased decision models that can arise due to the use of biased data. For instance, consider the 'gender' feature used in the decision trees shown in Figure 2. This feature could be considered a non-ethical feature to be used depending on the decision domain. Decision trees (or other symbolic models) explaining black-box models provide a means to identify such biases. Second, we investigate to apply our approach to explain CNNs in image classification. In such a setting, we foresee ontologies as the semantic layer able to bridge the gap between the pixel-based description of an image, and a more abstract description made by properties and objects; and the use of decision tree as explanations of different levels of abstractions of the CNNs layers. Finally, another area in which we would like to explore the usage of this approach is recommender systems. We

[8] `https://www.perspicuous-computing.science`

are currently looking into how to use this approach to explain recommendations generated by a neural collaborative filtering algorithm [14].

References

1. Alvarez-Melis, D., Jaakkola, T.S.: A causal framework for explaining the predictions of black-box sequence-to-sequence models. CoRR **abs/1707.01943** (2017)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: IJCAI. pp. 364–369 (2005)
3. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, New York, NY, USA (2003)
4. Baayen, R., Davidson, D., Bates, D.: 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
5. Buchanan, B.G., Shortliffe, E.H.: Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project. Addison-Wesley Longman Publishing Co., Inc. (1984)
6. Confalonieri, R., Eppe, M., Schorlemmer, M., Kutz, O., Peñaloza, R., Plaza, E.: Upward refinement operators for conceptual blending in the description logic $\mathcal{EL}^{++}$. Annals of Mathematics and Artificial Intelligence **82**(1), 69–99 (2018)
7. Craven, M.W., Shavlik, J.W.: Extracting tree-structured representations of trained networks. In: Proceedings of the 8th International Conference on Neural Information Processing Systems. pp. 24–30. NIPS'95, MIT Press, Cambridge, MA, USA (1995)
8. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. CoRR **abs/1702.08608** (2017)
9. Eleanor, Rosch, E., Carolyn, B., Mervis, C.B., Gray, W.D., Johnson, D.M., Penny, J.L., Boyes-Braem, P.: Basic objects in natural categories. Cognitive Psychology **8**, 382–439 (1976)
10. Frosst, N., Hinton, G.E.: Distilling a neural network into a soft decision tree. CoRR **abs/1711.09784** (2017)
11. d'Avila Garcez, A.S., Broda, K., Gabbay, D.M.: Symbolic knowledge extraction from trained neural networks: A sound approach. Artificial Intelligence **125**(1-2), 155–207 (2001)
12. Guarino, N., Oberle, D., Staab, S.: Handbook on Ontologies, chap. What is an Ontology?, pp. 1–17. International Handbooks on Information Systems, Springer, Berlin (2009)
13. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Comp. Surv. **51**(5), 1–42 (2018)
14. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proc. of the 26th International Conference on World Wide Web. pp. 173–182. WWW '17 (2017)
15. Hendricks, L.A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., Darrell, T.: Generating visual explanations. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. pp. 3–19. Springer International Publishing, Cham (2016)

16. Hoffman, R.R., Mueller, S.T., Klein, G., Litman, J.: Metrics for explainable AI: challenges and prospects. CoRR **abs/1812.04608** (2018)
17. Huysmans, J., Dejaeger, K., Mues, C., Vanthienen, J., Baesens, B.: An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. Decision Support Systems **51**(1), 141–154 (2011)
18. Kim, B., Rudin, C., Shah, J.: The bayesian case model: A generative approach for case-based reasoning and prototype classification. In: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2. pp. 1952–1960. NIPS'14, MIT Press, Cambridge, MA, USA (2014)
19. van der Laag, P.R., Nienhuys-Cheng, S.H.: Completeness and properness of refinement operators in inductive logic programming. The Journal of Logic Programming **34**(3), 201 – 225 (1998)
20. Laurence, S., Margolis, E.: Concepts and cognitive science. In: Margolis, E., Laurence, S. (eds.) Concepts: Core Readings, pp. 3–81. MIT Press (1999)
21. Lehmann, J., Hitzler, P.: Concept learning in description logics using refinement operators. Machine Learning **78**(1-2), 203–250 (2010)
22. Lipton, Z.C.: The mythos of model interpretability. Queue **16**(3), 30:31–30:57 (Jun 2018)
23. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. CoRR **abs/1706.07269** (2017)
24. Neuhaus, F.: What is an ontology? CoRR **abs/1810.09171** (2018), `http://arxiv.org/abs/1810.09171`
25. Odense, S., d'Avila Garcez, A.: Extracting m of n rules from restricted boltzmann machines. In: Lintas, A., Rovetta, S., Verschure, P.F., Villa, A.E. (eds.) Artificial Neural Networks and Machine Learning – ICANN 2017. pp. 120–127. Springer International Publishing, Cham (2017)
26. Park, D.H., Hendricks, L.A., Akata, Z., Schiele, B., Darrell, T., Rohrbach, M.: Attentive explanations: Justifying decisions and pointing to the evidence. CoRR **abs/1612.04757** (2016)
27. Parliament and Council of the European Union: General Data Protection Regulation (2016)
28. Piltaver, R., Luštrek, M., Gams, M., Martinčić-Ipšić, S.: What makes classification trees comprehensible? Expert Syst. Appl. **62**(C), 333–346 (Nov 2016)
29. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1. pp. 448–453. IJCAI'95, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1995)
30. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In: Proc. of the 22nd Int. Conf. on Knowledge Discovery and Data Mining. pp. 1135–1144. KDD '16, ACM (2016)
31. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: High-precision model-agnostic explanations. In: AAAI. pp. 1527–1535. AAAI Press (2018)
32. Rizzo, G., dAmato, C., Fanizzi, N., Esposito, F.: Tree-based models for inductive classification on the web of data. Journal of Web Semantics **45**, 1 – 22 (2017)
33. Samek, W., Wiegand, T., Müller, K.: Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models. CoRR **abs/1708.08296** (2017), `http://arxiv.org/abs/1708.08296`
34. Sánchez, D., Batet, M., Isern, D.: Ontology-based information content computation. Knowledge-Based Systems **24**(2), 297 – 303 (2011)
35. Towell, G.G., Shavlik, J.W.: Extracting refined rules from knowledge-based neural networks. Machine Learning **13**(1), 71–101 (Oct 1993)

36. Troquard, N., Confalonieri, R., Galliani, P., Peñaloza, R., Porello, D., Kutz, O.: Repairing Ontologies via Axiom Weakening. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence. pp. 1981–1988 (2018)
37. Wick, M.R., Thompson, W.B.: Reconstructive expert system explanation. Artificial Intelligence **54**(1-2), 33–70 (Mar 1992)
38. Zhang, J., Silvescu, A., Honavar, V.G.: Ontology-driven induction of decision trees at multiple levels of abstraction. In: Koenig, S., Holte, R.C. (eds.) Abstraction, Reformulation and Approximation, 5th International Symposium, SARA 2002, Kananaskis, Alberta, Canada, August 2-4, 2002, Proceedings. Lecture Notes in Computer Science, vol. 2371, pp. 316–323. Springer (2002)
39. Zhang, Y., Chen, X.: Explainable recommendation: A survey and new perspectives. CoRR **abs/1804.11192** (2018)
40. Zhou, Z.H., Jiang, Y., Chen, S.F.: Extracting symbolic rules from trained neural network ensembles. AI Communications **16**(1), 3–15 (2003)