# Eliminating The Impossible, Whatever Remains Must Be True

## On Extracting and Applying Background Knowledge In The Context Of Formal Explanations

**Jinqiang Yu**[1,4], **Alexey Ignatiev**[1], **Peter J. Stuckey**[1,4], **Nina Narodytska**[2], **Joao Marques-Silva**[3]

[1] Monash University, Melbourne, Australia
[2] VMWare Research, Palo Alto, USA
[3] IRIT, CNRS, Toulouse, France
[4] ARC Training Centre in OPTIMA, Melbourne, Australia
{jinqiang.yu,alexey.ignatiev,peter.stuckey}@monash.edu, nnarodytska@vmware.com, joao.marques-silva@irit.fr

**Abstract**

The rise of AI methods to make predictions and decisions has led to a pressing need for more explainable artificial intelligence (XAI) methods. One common approach for XAI is to produce a post-hoc explanation, explaining why a black box ML model made a certain prediction. Formal approaches to post-hoc explanations provide succinct reasons for *why* a prediction was made, as well as *why not* another prediction was made. But these approaches assume that features are independent and uniformly distributed. While this means that "why" explanations are correct, they may be longer than required. It also means the "why not" explanations may be suspect as the counterexamples they rely on may not be meaningful. In this paper, we show how one can apply background knowledge to give more succinct "why" formal explanations, that are presumably easier to interpret by humans, and give more accurate "why not" explanations. In addition, we show how to use existing rule induction techniques to efficiently extract background information from a dataset.

## 1 Introduction

Recent years have witnessed rapid advances in Artificial Intelligence (AI) and Machine Learning (ML) algorithms revolutionizing all aspects of human lives (LeCun, Bengio, and Hinton 2015; ACM 2018). An ever growing range of practical applications of AI and ML, on the one hand, and a number of critical issues observed in modern AI systems (e.g. decision bias (Angwin et al. 2016) and brittleness (Szegedy et al. 2014)), on the other hand, gave rise to the quickly advancing area of theory and practice of Explainable AI (XAI).

Several major approaches to XAI exist. Besides tackling XAI through computing *interpretable* ML models directly (Rudin 2019), or through the use of interpretable models for approximating complex *black-box* ML models (Ribeiro, Singh, and Guestrin 2016), the most prominent approach to XAI is to compute *post-hoc explanations* to ML predictions on demand (Lundberg and Lee 2017; Ribeiro, Singh, and Guestrin 2018). Prior work distinguishes post-hoc (*abductive*) explanations answering a *"why?"* question and (*contrastive*) explanations targeting a *"why not?"* question (Miller 2019). Heuristic approaches to post-hoc explainability are known to suffer from a number of fundamental explanation quality issues (Narodytska et al. 2019; Ignatiev, Narodytska, and Marques-Silva 2019c; Camburu et al. 2019; Ignatiev 2020), including the existence of out-of-distribution attacks (Slack et al. 2020). A promising alternative is formal explainability where explanations are computed as prime implicants of the decision function associated with ML predictions (Shih, Choi, and Darwiche 2018). Formal explanations have also been related with abductive reasoning (Ignatiev, Narodytska, and Marques-Silva 2019a,b).

Although provably correct and minimal, formal explanations have a few limitations. To provide provable correctness guarantees, formal approaches have to take into account the complete feature space assuming that the features are independent and uniformly distributed (Wäldchen et al. 2021). This makes a formal reasoner check all the combinations of feature values, including those that realistically can *never appear* in practice. This leads to unnecessarily long explanations that are hard for a human decision maker to interpret.

Motivated by this limitation, our work focuses on computing both abductive and contrastive formal explanations making use of background knowledge, and makes the following contributions. First, given a training data, we propose an efficient generic approach to extracting background knowledge in the form of highly accurate *if-then* rules, building on a recent formal method for learning decision sets (Ignatiev et al. 2021). Second, we propose a novel approach to computing formal explanations subject to background knowledge, *independent* of the nature of the background knowledge. Third, we prove theoretically that the use of background knowledge positively affects the quality of both abductive and contrastive explanations, thus, helping to build trust in the underlying AI systems. Finally, motivated by (Ignatiev 2020), we argue that background knowledge helps one assess the correctness of heuristic ML explainers more accurately since it blocks impossible combinations of features values.

## 2 Preliminaries

**SAT and MaxSAT.** Definitions standard in *propositional satisfiability* (SAT) and *maximum satisfiability* (MaxSAT) solving are assumed (Biere et al. 2021). SAT and MaxSAT formulas are assumed to be propositional. A propositional formula $\varphi$ is considered to be in *conjunctive normal form* (CNF) if it is a conjunction (logical *"and"*) of clauses, where a *clause* is a disjunction (logical *"or"*) of literals, and a

Table 1: Several examples extracted from *adult* dataset.

| Education | Status | Occupation | Relationship | Sex | Hours/w | Target |
|---|---|---|---|---|---|---|
| HighSchool | Married | Sales | Husband | Male | 40 to 45 | $\geq$ 50k |
| Bachelors | Married | Sales | Wife | Female | $\leq$ 40 | $\geq$ 50k |
| Masters | Married | Professional | Wife | Female | $\geq$ 45 | $\geq$ 50k |
| Masters | Married | Professional | Wife | Female | $\leq$ 40 | $\geq$ 50k |
| Dropout | Separated | Service | Not-in-family | Male | $\leq$ 40 | $<$ 50k |
| Dropout | Never-Married | Blue-Collar | Unmarried | Male | $\geq$ 45 | $\geq$ 50k |

*literal* is either a Boolean variable $b$ or its *negation* $\neg b$. Whenever convenient, a clause is treated as a set of literals. In the context of *unsatisfiable* formulas, i.e. when there is no way to assign the values 0 or 1 to all the variables of formula $\varphi$ s.t. $\varphi$ evaluates to 1, the maximum satisfiability problem is to find a truth assignment that maximizes the number of satisfied clauses. Hereinafter, we use a variant of MaxSAT called Partial (Unweighted) MaxSAT (Biere et al. 2021, Chapters 23 and 24). The formula $\varphi$ in Partial (Unweighted) MaxSAT is a conjunction of *hard* clauses $\mathcal{H}$, which must be satisfied, and *soft* clauses $\mathcal{S}$, which represent a preference to satisfy them, i.e. $\varphi = \mathcal{H} \land \mathcal{S}$. The aim is to find a truth assignment that satisfies all hard clauses while maximizing the total number of satisfied soft clauses.

Hereinafter, propositional formulas are applied for reasoning about the behavior of machine learning models used as well as to represent background knowledge constraints.

**Classification Problems.** Classification problems consider a set of classes $\mathcal{K} = \{c_1, c_2, \ldots, c_k\}$, and a set of features $\mathcal{F} = \{1, \ldots, m\}$. The value of each feature $i \in \mathcal{F}$ is taken from a domain $\mathcal{D}_i$, which can be integer, real-valued or Boolean. Therefore, the complete feature space is defined as $\mathbb{F} \triangleq \prod_{i=1}^{m} \mathcal{D}_i$. A concrete point in feature space is represented by $\mathbf{v} = (v_1, \ldots, v_m) \in \mathbb{F}$, where each $v_i \in \mathbf{v}$ is a constant taken by feature $i \in \mathcal{F}$. An *instance* or *example* is denoted by a specific point $\mathbf{v} \in \mathbb{F}$ in feature space and its corresponding class $c \in \mathcal{K}$, i.e. a pair $(\mathbf{v}, c)$ represents an instance. Moreover, the notation $\mathbf{x} = (x_1, \ldots, x_m)$ denotes an arbitrary point in feature space, where each $x_i \in \mathbf{x}$ is a variable taking values from its corresponding domain $\mathcal{D}_i$ and representing feature $i \in \mathcal{F}$.

A classifier defines a classification function $\tau : \mathbb{F} \to \mathcal{K}$. There are many ways to learn classifiers for a given dataset. In this paper, we consider: *decision lists* (DLs) (Rivest 1987; Clark and Niblett 1989), *boosted trees* (BTs) (Friedman 2001; Chen and Guestrin 2016), and *binarized neural networks* (BNNs) (Hubara et al. 2016).

**Example 1.** *Consider the data shown in Table 1. It represents a snapshot of instances taken from a simplified version*[1] *of the* adult *dataset (Kohavi 1996). Figure 1 illustrates DL and BT models trained for this dataset. Observe that for instance* $\mathbf{v} = \{$*Education = HighSchool, Status = Married, Occupation = Sales, Relationship = Husband, Sex = Male, Hours/w = 40 to 45*$\}$ *from Table 1, rule* $R_2$ *in the DL in Figure 1a predicts* $\geq 50k$*. Similarly, the sum of the weights (*$0.1063$*,* $0.0707$ *and* $-0.0128$ *in the* $3$ *trees, respectively) for prediction* $\geq 50k$ *is positive (*$0.1642$*) in the BT in Figure 1b, and so the BT model also predicts* $\geq 50k$ *for the aforementioned instance* $\mathbf{v}$*.*

[1]For simplicity, the running example used throughout the text will correspond to a *simplified* version of the *adult* dataset (Kohavi 1996), where some of the features are dropped.

**Interpretability and Explanations.** Interpretability is not formally defined since it is a subjective concept (Lipton 2018). In this paper, we define interpretability as the conciseness of the computed explanations for an ML model to justify a provided prediction. The definition of explanation for an ML model is built on earlier work (Shih, Choi, and Darwiche 2018; Ignatiev, Narodytska, and Marques-Silva 2019a; Darwiche and Hirth 2020; Audemard, Koriche, and Marquis 2020; Marques-Silva and Ignatiev 2022), where explanations are equated with *abductive explanations* (AXps), which are subset-minimal sets of features sufficing to explain a given ML prediction. Concretely, given an instance $\mathbf{v} \in \mathbb{F}$ and a computed prediction $c \in \mathcal{K}$, i.e. $\tau(\mathbf{v}) = c$, an AXp is a subset-minimal set of features $\mathcal{X} \subseteq \mathcal{F}$, such that

$$\forall(\mathbf{x} \in \mathbb{F}). \bigwedge\nolimits_{i \in \mathcal{X}} (x_i = v_i) \rightarrow (\tau(\mathbf{x}) = c) \tag{1}$$

Abductive explanations are also prime implicants of the decision predicate $\tau_c$ and hence a *prime implicant* (PI) explanation is another name for an AXp.

**Example 2.** *Consider the models in Figure 1 and instance* $\mathbf{v}$ *from Example 1. By examining the DL model, specifying Education = HighSchool, Status = Married, Occupation = Sales, and Relationship = Husband guarantees that any compatible instance is classified by* $R_2$ *independent of the values of other features, i.e. Sex and Hours/w. Similarly, the prediction of an instance is guaranteed to be* $\geq 50k$ *in Figure 1b as long as the feature values above are used, since the sum of weights is promised to be* $0.1063 + 0.0707 + -0.0128 = 0.1642$ *for class* $\geq 50k$*. Therefore, the (only) AXp* $\mathcal{X}$ *for the prediction of* $\mathbf{v}$ *is* $\{$*Education, Status, Occupation, Relationship*$\}$ *in both models.*

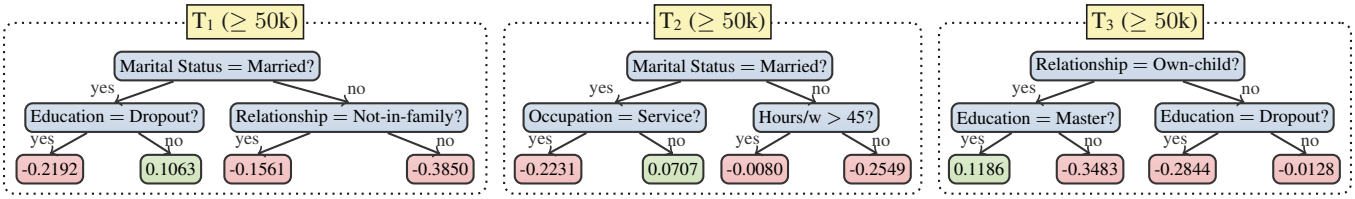
We also consider *contrastive explanations* (CXps) defined as subset-minimal sets of features that are necessary to change the prediction if the features of a CXp are allowed to take arbitrary values from their domains. Formally and following (Ignatiev et al. 2020), a CXp for prediction $\tau(\mathbf{v}) = c$ is defined as a minimal subset $\mathcal{Y} \subseteq \mathcal{F}$ s.t.

$$\exists(\mathbf{x} \in \mathbb{F}). \bigwedge\nolimits_{i \notin \mathcal{Y}} (x_i = v_i) \land (\tau(\mathbf{x}) \neq c) \tag{2}$$

**Example 3.** *Consider the setup of Example 2. Given either model,* $\mathcal{Y} = \{$*Occupation*$\}$ *is a CXp for instance* $\mathbf{v}$ *because the prediction for* $\mathbf{v}$ *can be changed if feature 'Occupation' is allowed to take another value, e.g. if the value is changed to 'Service'. Similarly, changing the value of feature 'Occupation' to 'Service' triggers that the weights in the* $3$ *trees become* $0.1063$*,* $-0.2231$ *and* $-0.0128$*. Therefore, the total weight is* $-0.0982$*, i.e. the prediction is changed. By further examining the two models, other subsets of features can be identified as CXps for* $\mathbf{v}$*. The set of CXps is* $\mathbb{Y} = \{\{$*Education*$\}, \{$*Status*$\}, \{$*Occupation*$\}, \{$*Relationship*$\}\}$*, while the set of AXps demonstrated in Example 2 is* $\mathbb{X} = \{\{$*Education, Status, Occupation, Relationship*$\}\}$*.*

| | | | | |
|---|---|---|---|---|
| $R_0$: | IF | Education = Dropout | THEN | Target $<$ 50k |
| $R_1$: | ELSE IF | Occupation = Service | THEN | Target $<$ 50k |
| $R_2$: | ELSE IF | Status = Married $\wedge$ Relationship = Husband | THEN | Target $\geq$ 50k |
| $R_3$: | ELSE IF | Status = Married $\wedge$ Relationship = Wife | THEN | Target $\geq$ 50k |
| $R_{DEF}$: | ELSE | | THEN | Target $<$ 50k |

(a) Decision list.




(b) Boosted tree (Chen and Guestrin 2016) consisting of 3 trees with the depth of each tree at most 2.

Figure 1: Example DL and BT models trained on the well-known *adult* classification dataset.

(Ignatiev et al. 2020) builds on the seminal work of Reiter (Reiter 1987) to establish a minimal hitting set (MHS) duality relationship between AXps and CXps, i.e. each CXp *minimally hit* every AXp, and vice-versa.

**Example 4.** *Observe how the MHS duality holds for the sets of AXps $\mathbb{X}$ and the set of CXps $\mathbb{Y}$ shown in Example 3. The only AXp minimally hits all the CXps and vice versa.*

There is a growing body of recent work on formal explanations (Marques-Silva et al. 2020, 2021; Izza and Marques-Silva 2021; Ignatiev and Marques-Silva 2021; Arenas et al. 2021; Wäldchen et al. 2021; Darwiche and Marquis 2021; Malfa et al. 2021; Boumazouza et al. 2021; Blanc, Lange, and Tan 2021; Izza, Ignatiev, and Marques-Silva 2022; Gorji and Rubin 2022; Ignatiev et al. 2022; Huang et al. 2022; Marques-Silva and Ignatiev 2022; Amgoud and Ben-Naim 2022; Ferreira et al. 2022; Arenas et al. 2022).

## 3 Extracting Background Knowledge

Recent work (Gorji and Rubin 2022) argues that background knowledge is helpful in the context of formal explanations. If identified, background knowledge may help forbid some of the combinations of feature values that would otherwise have to be taken into account by a formal reasoner, thus, slowing the reasoner down and making the explanations unnecessarily long. But the question of how such knowledge can be obtained in an automated way remains open.

**Example 5.** *Assume that Table 1 represents* trustable *information. The following two rules can be extracted:*

- IF *Relationship* $=$ *Husband* THEN *Status* $=$ *Married*
- IF *Relationship* $=$ *Wife* THEN *Status* $=$ *Married*

*These rules may be used to discard feature* Status *when computing explanations as long as* Relationship *equals either* Husband *or* Wife *because of the implications identified.*

We describe the MaxSAT approach to extract background knowledge representing implicit relations between features of a dataset if the dataset is assumed to be trustable. It builds on the recent two-stage approach (Ignatiev et al. 2021) to learning smallest size decision sets. Concretely, we apply the first stage of (Ignatiev et al. 2021) which enumerates individual decision rules given a dataset, using MaxSAT.

Without diving into the details, the idea of (Ignatiev et al. 2021) is as follows. Given training data $\mathcal{E}$ and target class $c \in \mathcal{K}$, a MaxSAT solver is invoked multiple times, each producing a unique subset-minimal (irreducible) rule in the form of *"IF* antecedent *THEN* prediction $c$*"*, where the antecedent is a set of feature values. The MaxSAT solver is fed with various CNF constraints and an objective function targeting rule size minimization. The approach also detects and blocks *symmetric rules*, i.e. those that do not contribute new information to the rule-based representation of class $c \in \mathcal{K}$.

We can modify the MaxSAT approach outlined above to learning background knowledge in the form of decision rules, i.e. identifying the dependency of a feature $i \in \mathcal{F}$ on other features $j \in \mathcal{F} \setminus \{i\}$. For this, we need to discard the prediction column from the dataset $\mathcal{E}$ and instead focus on a feature $i \in \mathcal{F}$, consider some of its values $v_{ij} \in \mathcal{D}_i$ and "pretend" to compute decision rules for a "fake class" $x_i = v_{ij}$. Thanks to the properties of the approach of (Ignatiev et al. 2021), all the rules computed are guaranteed to be subset-minimal and to respect training data $\mathcal{E}$. Once all the rules for feature $i \in \mathcal{F}$ and value $v_{ij} \in \mathcal{D}_i$ are computed, the same exercise can be repeated for all the values in $\mathcal{D}_i \setminus \{v_{ij}\}$ but, more importantly, all the other features.

**Example 6.** *Consider again Table 1. The two rules shown in Example 5 are computed by our rule learning approach if we focus on feature* Status*. The following two rules can be extracted when feature* Relationship *is focused on instead:*

- IF *Status* $=$ *Married* $\wedge$ *Sex* $=$ *M.* THEN *Rel.* $=$ *Husband*
- IF *Status* $=$ *Married* $\wedge$ *Sex* $=$ *F.* THEN *Rel.* $=$ *Wife*

**Duplicate Rules.** As mentioned above, all rules generated with the MaxSAT approach of (Ignatiev et al. 2021) are guaranteed to be subset-minimal. Furthermore, none of the rules enumerated is symmetric with another rule if considered in the *if-then* form. However, when the rules are

Algorithm 1: Rule Extraction

**Input**: Dataset $\mathcal{E}$, extraction limit $\lambda$
**Output**: Rules $\varphi$

$\mathcal{E}_f, \mathcal{F} \leftarrow \texttt{DropClass}(\mathcal{E}), \texttt{ExtractFeatures}(\mathcal{E})$
$\varphi, B \leftarrow \emptyset, \emptyset$ *# to extract and block rules, resp.*
**for** $i \in \mathcal{F}$ **do**
  **for** rule $\in \texttt{EnumerateRules}(\mathcal{E}_f, i, B)$ **do**
    **if** $\texttt{limit}(\text{rule}, \lambda)$ is $true$ **then**
      **break**
    $\varphi \leftarrow \varphi \cup \text{rule}$
  $B \leftarrow \varphi$
**return** $\varphi$

treated as clauses, i.e. a disjunction of Boolean literals, some rules may duplicate the other. Indeed, recall that a rule of size $k \leq |\mathcal{F}|$ is of the form $(f_1 \wedge \ldots \wedge f_{k-1}) \rightarrow f_k$ where each $f_i$ represents a literal $(x_i = v_{i_{j_i}})$, $i \in \mathcal{F}$ and $v_{i_{j_i}} \in \mathcal{D}_i$. Clearly, this same proposition can be equivalently represented as a clause $(\neg f_1 \vee \ldots \vee \neg f_{k-1} \vee f_k)$. Observe that the same clause can be used to represent another rule $(f_1 \wedge \ldots \wedge f_{k-2} \wedge \neg f_k) \rightarrow \neg f_{k-1}$, which can thus be seen as symmetric in the *clausal form*. This way, a clause of size $k$ represents $k$ possible rules. However, due to symmetry, it suffices to compute only one of them and block all the "duplicates" by adding its clausal representation to the MaxSAT solver. This novel symmetry breaking mechanism significantly improves the scalability of our approach.

**Example 7.** *Consider a rule* { *IF Status* = *Married* $\wedge$ *Sex* = *Male THEN Relationship* = *Husband* } *computed for feature Relationship. This rule is represented as a clause*

(*Status* $\neq$ *Married* $\vee$ *Sex* $\neq$ *M.* $\vee$ *Relationship* = *Husband*)

*There are two duplicates in other contexts:*

- IF *Status* = *Married* $\wedge$ *Rel.* $\neq$ *Husband* THEN *Sex* = *F.*
- IF *Sex* = *M.* $\wedge$ *Rel.* $\neq$ *Husband* THEN *Status* $\neq$ *Married*

**Extraction limit.** Even if we remove duplicate rules, there can still be many rules to enumerate for an entire dataset. Many of them will never, or only rarely, contribute to reducing the size of explanations of the classifier. Extracting these *low value* rules is unnecessary in the rule extracting process. In practice, we noticed that some rules (e.g. long rules or rules having a low support) never contribute to explanation reduction. Hence, we apply an *extraction limit* to prevent exhaustive rule enumeration, which enables us to focus only on *most useful* rules. Here, extraction limit can be a restriction of a user's choice, e.g. a total extraction runtime, a limit on the number of rules, rule support or size, etc.

A high-level view on the overall rule extraction approach is provided in Algorithm 1. Initially, the class column from the original dataset $\mathcal{E}$ is dropped and the features $\mathcal{F}$ in $\mathcal{E}$ are acquired. For each feature $i \in \mathcal{F}$, the algorithm enumerates the decision rules targeting $i$ until the extraction limit is met or no more rules can be found. The rules previously learned are blocked in the clausal form to avoid computing their duplicates. Finally, the algorithm returns the rules extracted.

Our approach computes only rules that are perfectly consistent with the *known* data, which makes sense if the data is extensive and trustworthy. In practical settings, however, some of the data are unknown, i.e. the rules computed may be inconsistent with unseen parts of the feature space $\mathbb{F}$. If testing and validation data are available, then the rules can be tested against them. We can then exclude the rules that are not *sufficiently* accurate wrt. test and/or validation data.

## 4 Knowledge-Assisted Explanations

We assume the obtained background knowledge can be represented as a formula $\varphi$. Under that assumption, (Gorji and Rubin 2022) proposes to compute AXps for positive predictions of a Boolean classifier $\tau : \mathbb{F} \rightarrow \{0, 1\}$ taking into account constraints $\varphi$. Observe that formula $\varphi$ can be seen as representing a predicate $\varphi : \mathbb{F} \rightarrow \{0, 1\}$, the truth value of which, i.e. $\varphi(\mathbf{x})$, can be tested for an instance $\mathbf{v} \in \mathbb{F}$. The approach of (Gorji and Rubin 2022) relies on *compiling* a Boolean classifier $\tau(\mathbf{x})$ into a *tractable* representation (Shih, Choi, and Darwiche 2018) and proposes to compute an AXp $\mathcal{X} \subseteq \mathcal{F}$ for prediction $\tau(\mathbf{v}) = 1$, $\mathbf{v} \in \mathbb{F}$, subject to constraints $\varphi$ as a prime implicant of $[\varphi(\mathbf{x}) \rightarrow \tau(\mathbf{x}) = 1]$.

Observe that we can generalize this idea to the context of computing formal AXps and CXps for *any classifier* that admits a logical representation suitable for making reasoning oracle calls wrt. formulas (1) and (2). Namely, given a prediction $\tau(\mathbf{x}) = c$, $\mathbf{v} \in \mathbb{F}$, $c \in \mathcal{K}$, an abductive explanation $\mathcal{X} \subseteq \mathcal{F}$ subject to background knowledge $\varphi$ is such that:

$$\forall(\mathbf{x} \in \mathbb{F}). \bigwedge\nolimits_{j \in \mathcal{X}} (x_j = v_j) \rightarrow [\varphi(\mathbf{x}) \rightarrow (\tau(\mathbf{x}) = c)] \quad (3)$$

More importantly, the same can be done with respect to contrastive explanations. Given a prediction $\tau(\mathbf{x}) = c$, $\mathbf{v} \in \mathbb{F}$, $c \in \mathcal{K}$, a contrastive explanation $\mathcal{Y} \subseteq \mathcal{F}$ subject to background knowledge $\varphi$ is such that the following holds:

$$\exists(\mathbf{x} \in \mathbb{F}). \bigwedge\nolimits_{i \notin \mathcal{Y}} (x_i = v_i) \wedge [\varphi(\mathbf{x}) \wedge (\tau(\mathbf{x}) \neq c)] \quad (4)$$

Note that (3) and (4) are the negation of each other, i.e. a subset of features $\mathcal{Y} \subseteq \mathcal{F}$ *is* a CXp for prediction $\tau(\mathbf{x}) = c$ iff $\mathcal{X} = \mathcal{F} \setminus \mathcal{Y}$ *is not* an AXp. This means when dealing with either AXps or CXps, one can reason about (un)satisfiability of formula $\bigwedge_{i \in \mathcal{Z}} (x_i = v_i) \wedge [\varphi(\mathbf{x}) \wedge (\tau(\mathbf{x}) \neq c)]$ with $\mathcal{Z}$ being either $\mathcal{X}$ or $\mathcal{F} \setminus \mathcal{Y}$ depending on the kind of target explanation. Therefore, if background knowledge $\varphi$ is a *conjunction* of constraints, e.g. rules, we can integrate them in the existing formal explainability setup of (Ignatiev, Narodytska, and Marques-Silva 2019a) with no additional overhead.

Following (Ignatiev et al. 2020) and applying the same arguments, an immediate observation to make is that in the presence of background knowledge, the minimal hitting set duality between AXps and CXps holds:

**Proposition 1.** *Let* $\mathbf{v} \in \mathbb{F}$ *be an instance such that* $\tau(\mathbf{v}) = c$, $c \in \mathcal{K}$, *and background knowledge* $\varphi$ *is compatible with* $\mathbf{v}$. *Then any AXp* $\mathcal{X}$ *for prediction* $\tau(\mathbf{v}) = c$ *minimally hits any CXp for this prediction, and vice versa.*

Proposition 1 enables us to apply algorithms originally studied in the context of over-constrained systems (Bailey and Stuckey 2005; Liffiton and Sakallah 2008; Belov, Lynce, and Marques-Silva 2012; Marques-Silva et al. 2013; Mencia, Previti, and Marques-Silva 2015; Ignatiev et al. 2015;

| | | | | |
|---|---|---|---|---|
| $R_0$: | IF | Status = Married | THEN | Target $\geq 50k$ |
| $R_1$: | ELSE IF | Sex = Male $\land$ Relationship $\neq$ Husband | THEN | Target $< 50k$ |
| $R_{DEF}$: | ELSE | | THEN | Target $\geq 50k$ |

Figure 2: A DL for selected examples of *adult* dataset.

Liffiton et al. 2016; Bendík, Cerná, and Benes 2018) to explore all AXps and CXps for ML predictions. In particular, the existing explanation extraction and enumeration algorithms (Ignatiev, Narodytska, and Marques-Silva 2019a; Ignatiev et al. 2020) can be readily applied by taking into account background knowledge, as shown in (3) and (4).

Gorji et al. (Gorji and Rubin 2022) proved that subset-minimal AXps computed subject to additional constraints for Boolean classifiers tend to be smaller than their unconstrained "counterparts". The rationale is that when additional constraints are imposed, some of the features $i \in \mathcal{F}$ may be dropped from an AXp because the equalities $x_i = v_i$ falsify the constraints, i.e. they represent data instances that are *not permitted* by the constraints. Based on their result, the following generalization can be proved to hold:

**Proposition 2.** *Consider $\mathbf{v} \in \mathbb{F}$ such that $\tau(\mathbf{v}) = c$, $c \in \mathcal{K}$, and background knowledge $\varphi$ is compatible with $\mathbf{v}$. Then for any subset-minimal AXp $\mathcal{X} \subseteq \mathcal{F}$ for prediction $\tau(\mathbf{v}) = c$, there is a subset-minimal AXp $\mathcal{X}' \subseteq \mathcal{F}$ for $\tau(\mathbf{v}) = c$ subject to background knowledge $\varphi$ such that $\mathcal{X}' \subseteq \mathcal{X}$.*

**Remark 1.** *The opposite, i.e. that given AXp $\mathcal{X}'$ subject to background knowledge $\varphi$, there must exist a subset-minimal AXp $\mathcal{X} \supseteq \mathcal{X}'$ without knowledge $\varphi$, in general does not hold.*

**Example 8.** *Consider the DL in Figure 2. Given an instance $\mathbf{v} = \{$Education = Dropout, Status = Separated, Occupation = Service, Relationship = Not-in-Family, Sex = Male, Hours/w = $\leq 40\}$, the prediction enforced by $R_1$ is $\leq 50k$ and the AXp is $\mathcal{X} = \{$Status, Relationship, Sex$\}$. Let a single constraint $\varphi$ be $\{$Sex = Male $\land$ Relationship = Not-in-Family $\rightarrow$ Status = Separated$\}$. Feature 'Status' can be dropped because the constraint $\varphi$ ensures it to be set to the "right value" if the other two features are set as required, and hence $R_0$ is guaranteed not to fire. Thus, we can compute a smaller AXp $\mathcal{X}' = \{$ Relationship, Sex $\}$.*

While using background knowledge $\varphi$ pays off in terms of interpretability of abductive explanations, this cannot be said wrt. contrastive explanations. Surprisingly and as the following result proves, background knowledge can only contribute to increase the size of contrastive explanations.

**Proposition 3.** *Consider $\mathbf{v} \in \mathbb{F}$ such that $\tau(\mathbf{v}) = c$, $c \in \mathcal{K}$, and background knowledge $\varphi$ is compatible with $\mathbf{v}$. Then for any subset-minimal CXp $\mathcal{Y}' \subseteq \mathcal{F}$ for prediction $\tau(\mathbf{v}) = c$ subject to knowledge $\varphi$, there is subset-minimal CXp $\mathcal{Y} \subseteq \mathcal{F}$ is a CXp for prediction $\tau(\mathbf{v}) = c$ such that $\mathcal{Y}' \supseteq \mathcal{Y}$.*

**Remark 2.** *The reverse direction: given a CXp $\mathcal{Y}$ generated without using background knowledge, there must exist a CXp $\mathcal{Y}' \supseteq \mathcal{Y}$ using background knowledge, does not hold.*

One may wonder then why background knowledge is useful when computing CXps. The reason is that the CXps generated using background knowledge are *correct* under the assumption that the background knowledge describes the actual relationships between features. On the contrary, CXps generated without using background knowledge are *only correct* under the assumption that every combination of feature values is possible, i.e. all features are independent and their values are uniformly distributed across the feature space, which hardly ever occurs in practice.

**Example 9.** *Consider the setup of Example 8. Observe that a CXp for the prediction is $\mathcal{Y} = \{$ Status $\}$. Its correctness relies on the fact that changing Status to Married changes the prediction to $\geq 50k$. But given the background knowledge $\varphi$, this is clearly erroneous. Since the other fixed features in instance $\mathbf{v}$ are $\{$ Sex = Male, Education = Dropout, Occupation = Service, Relationship = Not-in-family, Hours/w = $\leq 40$ $\}$, the modification is inconsistent with knowledge $\varphi$. This demonstrates the weakness of CXps as they rely on the assumption that any tuple of feature values in $\mathbb{F}$ is possible. Applying constraint $\varphi$ leads to a larger CXp $\mathcal{Y}' \triangleq \{$ Status, Relationship $\}$. This clearly does allow the prediction to change and it is compatible with $\varphi$.*

## 5 Experimental Results

Here we provide a summary of the experimental results. The full experimental analysis can be found in (Yu et al. 2022).

**Setup and Prototype Implementation.** The experiments were run on an Intel Xeon 8260 CPU running Ubuntu 20.04.2 LTS, with a memory limit of 8 GByte. A prototype of the approach to extracting background knowledge and computing AXps and CXps applying background knowledge was developed as a set of Python scripts.[2] The implementation of knowledge extraction builds on (Ignatiev et al. 2021) and extensively uses modern SAT technology (Ignatiev, Morgado, and Marques-Silva 2018, 2019). Also, explanation enumeration for DLs and BNNs makes use of the SAT technology while for BTs we apply satisfiability modulo theories (SMT) solvers (Gario and Micheli 2015).

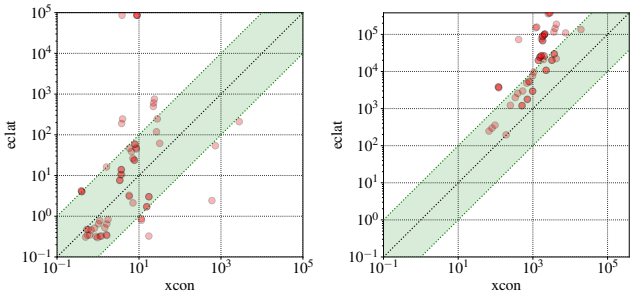
A few words should be said about the competition considered. First, we compared our knowledge extraction approach to the Apriori and Eclat algorithms (Agrawal and Srikant 1994; Zaki et al. 1997). In our experiments, these algorithms behave almost identically with Eclat solving one more instance; as a result, we use Eclat as the best competitor. When running Eclat, we apply the same setup as used for our approach. Finally, heuristic explainers are represented by LIME (Ribeiro, Singh, and Guestrin 2016), SHAP (Lundberg and Lee 2017), and Anchor (Ribeiro, Singh, and Guestrin 2018) in their default configurations.

**Datasets.** The benchmarks considered include a selection of datasets publicly available from UCI Machine Learning Repository (Dua and Graff 2017) and Penn Machine Learning Benchmarks (Olson et al. 2017). In total, 24 datasets are selected. Whenever applicable, numeric features in all benchmarks were quantized into 4, 5, or 6 intervals. Therefore, the total number of quantized datasets considered is 62.

[2] https://github.com/jinqiang-yu/xcon

Table 2: Average accuracy of individual rules over test data.

| | $\text{rule}_1$ | $\text{rule}_2$ | $\text{rule}_3$ | $\text{rule}_4$ | $\text{rule}_5$ | $\text{rule}_{all}$ |
|---|---|---|---|---|---|---|
| Accuracy (%) | 99.22 | 99.35 | 99.29 | 99.16 | 99.06 | 99.08 |




(a) Runtime comparison. (b) Number of rules comparison.

Figure 3: *Eclat* vs. *xcon* – performance comparison.

**Machine Learning Models.** We used CN2 (Clark and Niblett 1989) to train the DL models studied. BTs were computed by XGBoost (Ribeiro, Singh, and Guestrin 2016) s.t. each class is represented by 25 trees of depth 3. BNNs were trained by PyTorch (Paszke et al. 2019). Three configurations of hidden layers[3] were used when training BNNs to achieve sufficient test accuracy. As usual, each of the 62 datasets was randomly split into 80% of training and 20% of test data, respectively. The average test accuracy of the DL, BT, and BNN models was 76.47%, 76.17%, and 80.31%.

## 5.1 Knowledge Extraction

Knowledge extraction is tested using 5-fold cross validation. The average accuracy of each rule is measured as the proportion of test instances that violate that rule, averaged over the folds. We consider rules of length 1 to 5, and extracting all possible rules (*all*). Table 2 compares the average accuracy of the background knowledge extracted. The average accuracy of *all rules* and $rules_s$, $s \in \{1, \ldots, 5\}$, exceeds 99%.

Additionally, we compare the overall performance of exhaustive rule extraction against rule extraction with the size limit 5. On average, exhaustive (limited, resp.) rule enumeration ends up computing 2116.29 (1964.24, resp.) rules per dataset. We also compare *xcon* against Eclat in terms of the overall performance. For a fair comparison, we set Eclat to extract only rules of confidence 100%, i.e. all the rules extracted are perfectly consistent with the *known* data. Figure 3a compares the runtime of rule extraction and the number of extracted rules of size up to 5 across all 5 train-test pairs. Clearly *xcon* can extract rules on par with Eclat, and is able to tackle all 62 datasets, while Eclat fails on 4.

Figure 3b shows the number of extracted rules for the 58 datasets solved by both approaches. Eclat always extracts more rules because it uses a less expressive language, i.e. it cannot use the *negation* of feature literals.

[3]The 3 configurations are classified as *small*, *medium* and *large*. The size of the hidden layers of these 3 configurations is as follows: large: (64, 32, 24, 2); medium: (32, 16, 8, 2); small: (10, 5, 5, 2).

Table 3: Average runtime per explanation.

| Model | Runtime per explanation (ms) | | | | | | |
|---|---|---|---|---|---|---|---|
| | $\text{xcon}_{axp}$ | $\text{xcon}^r_{axp}$ | $\text{xcon}_{cxp}$ | $\text{xcon}^r_{cxp}$ | LIME | SHAP | Anchor |
| DL | 2 | 1 | 1 | 2 | 3755 | 42 555 | 3800 |
| BT | 80 | 82 | 97 | 151 | 98 | 6 | 351 |
| BNN | 196 | 152 | 179 | 199 | 15 607 | 183 058 | 11 384 |

## 5.2 Knowledge-Assisted Explanations

Let us evaluate the benefit of computing formal explations using background knowledge, once more restricting to background rules of size at most 5. For each of the 62 datasets, we selected all *test* instances and enumerated 20 *smallest size* AXps or CXps for each such instance. Let $xcon_*$ s.t. $* \in \{dl, bt, bnn\}$ denote the approaches for formally explaining DL, BT, and BNN models, and $xcon^r_*$ is the approach taking into account background knowledge.

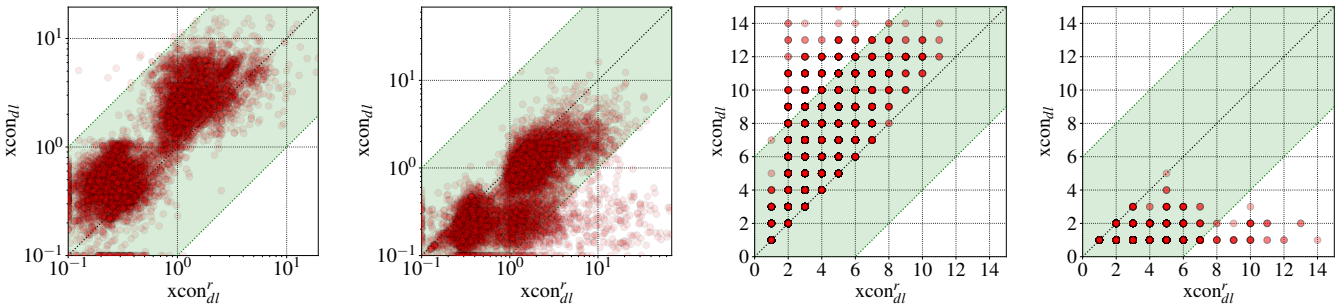
**Scalability.** Figures 4a and 4b compare the average runtime for computing a single AXp or CXp for the DL models. Clearly background knowledge quickens AXp explanation, while slowing CXp explanation. The slowdown for CXps is caused by the fact that the CXps get much larger, which requires a larger number of oracle calls. For BTs and BNNs the use of background knowledge neither substantially improves nor degrades the computation of AXps or CXps.

**Explanation Quality.** The change of smallest size of AXps and CXps in an instance for DLs is shown in Figure 4c, and 4d. Clearly background knowledge substantially reduces the size of AXps, and substantially increases the size of CXps. Similar results arise for BTs and BNNs. The experiments above were repeated using background knowledge extracted with Eclat. The results are similar.

## 5.3 Formal vs. Heuristic Explanations

Following (Ignatiev, Narodytska, and Marques-Silva 2019a; Narodytska et al. 2019; Ignatiev 2020), we apply formal explanations to assess the runtime and explanation quality for the heuristic approaches LIME, SHAP, and Anchor. The idea is to show the importance of trustable background knowledge when targeting a more accurate quality assessment.

**Scalability.** Table 3 illustrates the runtime of a single explanation extraction for a data instance across the 62 datasets performed by LIME, SHAP, Anchor, $xcon_*$, and $xcon^r_*$. Here, $xcon^r_*$ and $xcon_*$ represent the proposed approach to computing AXps or CXps with/without background knowledge, s.t. $* \in \{axp, cxp\}$. Observe that both $xcon_*$ and $xcon^r_*$ outperform LIME and Anchor for all the 3 models, explaining a data instance in a fraction of a second. LIME and Anchor are 1-2 orders of magnitude slower for DL and BNN models, while LIME outweighs Anchor when generating explanations for BTs. The worst performance for DL and BNN models is demonstrated by SHAP while, surprisingly, SHAP outperforms the other competitors for BTs models.




(a) AXp runtime for DLs. (b) CXp runtime for DLs. (c) AXp size for DLs. (d) CXp size for DLs.

Figure 4: Impact of *xcon* rules on runtime (ms) and explanation size for DLs.

Table 4: Average correctness of LIME, SHAP and Anchor.

| Explainer | Correctness (%) | | | | | |
|---|---|---|---|---|---|---|
| | Without knowledge | | | With knowledge | | |
| | DL | BT | BNN | DL | BT | BNN |
| LIME | 6.06 | 38.26 | 8.2 | 31.06 | 60.63 | 47.88 |
| SHAP | 49.47 | 72.89 | 58.89 | 91.72 | 93.75 | 95.0 |
| Anchor | 24.03 | 13.85 | 6.57 | 73.85 | 73.0 | 70.1 |

**Correctness.** The average correctness of computed explanations is shown in Table 4. [4] Here, an explanation is said to be correct if it answers a *"why"* question and satisfies (1) (or (3) in the presence of background knowledge) or it answers a *"why not"* question and satisfies (2) (or (4) in the presence of background knowledge). Table 4 shows that the average correctness is higher when background knowledge is applied as the number of features required in a minimal correct explanation answering a *"why"* question can drop, which is demonstrated in Section 5.2. However, heuristic approaches are not able to achieve 100% correctness.

Heuristic explainers consistently demonstrate low correctness when no background knowledge is applied, which confirms the earlier results of (Ignatiev 2020). This changes dramatically when we apply the background knowledge because some of the counterexamples invalidating heuristic explanations are forbidden by the knowledge extracted. Assuming that this knowledge is valid, these correctness results better reflect the reality and so are more trustable.

## 6 Related Work

Many methods for extracting knowledge from a dataset of rules exist (Hipp, Güntzer, and Nakhaeizadeh 2000; Zhang and Zhang 2002; Agrawal and Srikant 1994; Zaki et al. 1997; Izza et al. 2020; Belaid, Bessiere, and Lazaar 2019). For use as background knowledge, we aim at very high confidence in the rules, ideally they should be *completely* valid for the feature space. Traditional rule mining is more interested in rules with high support and less focused on validity, although it can be adapted to this case (see our experimental results above). Although the explanation methods we apply in the presence of background knowledge are agnostic about where it comes from, the motivation for our rule extraction method is twofold: (1) the rules are computed in a clausal form and (2) their high quality is guaranteed by the use of the strict optimization problem formulation.

The most prominent approaches to post-hoc explainability are of heuristic nature (Ribeiro, Singh, and Guestrin 2016; Lundberg and Lee 2017; Ribeiro, Singh, and Guestrin 2018) and based on sampling in the vicinity of the instances being explained. None of these approaches can handle background knowledge. Approaches to formal explainability are represented by compilation of classifiers into tractable representations (Shih, Choi, and Darwiche 2018) and reasoning-based explanation approaches (Ignatiev, Narodytska, and Marques-Silva 2019a; Marques-Silva and Ignatiev 2022). The closest related work is (Gorji and Rubin 2022). Based on compilation of a binary classifier into a binary decision diagram (BDD), it conjoins concocted background knowledge to give more succinct *"why"* explanations for the classifier. This approach is restricted to much smaller examples than we consider here, since the compilation of a classifier into a BDD tends to explode with the feature space. The SAT and SMT based approaches to explanation we use are far more scalable. Finally, we consider a much broader class of classifiers, and also examine *"why not"* explanations and how they can be improved by using background knowledge.

## 7 Conclusions

Using background knowledge is highly advantageous for producing formal explanations of machine learning models. For abductive explanations (AXps), the use of background knowledge substantially shortens explanations, making them easier to understand, *and* improves the speed of producing explanations. For contrastive explanations (CXps), while the background knowledge lengthens them and may increase the time required to generate an explanation, the resulting explanations are far more useful since they do not rely on the (usually unsupportable) assumption that all tuples in the feature space are possible. Furthermore and as this paper shows, background knowledge can be applied in the context of heuristic explanations when an accurate analysis of their correctness is required.

[4] LIME/SHAP assign weights to *all the features*. We use only those whose weight contributes to the decision made based on sign.

## Acknowledgments

This research was partially funded by the Australian Government through the Australian Research Council Industrial Transformation Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), Project ID IC200100009. This work was also partially supported by the AI Interdisciplinary Institute ANITI, funded by the French program "Investing for the Future – PIA3" under Grant agreement no. ANR-19-PI3A-0004, and by the H2020-ICT38 project COALA "Cognitive Assisted agile manufacturing for a Labor force supported by trustworthy Artificial intelligence".

## References

ACM. 2018. Fathers of the Deep Learning Revolution Receive ACM A.M. Turing Award. http://tiny.cc/9plzpz.

Agrawal, R.; and Srikant, R. 1994. Fast algorithms for mining association rules. In *VLDB*, 487–499.

Amgoud, L.; and Ben-Naim, J. 2022. Axiomatic Foundations of Explainability. In Raedt, L. D., ed., *IJCAI*, 636–642.

Angwin, J.; Larson, J.; Mattu, S.; and Kirchner, L. 2016. Machine Bias. http://tiny.cc/dd7mjz.

Arenas, M.; Baez, D.; Barceló, P.; Pérez, J.; and Subercaseaux, B. 2021. Foundations of Symbolic Languages for Model Interpretability. In *NeurIPS*.

Arenas, M.; Barceló, P.; Romero, M.; and Subercaseaux, B. 2022. On Computing Probabilistic Explanations for Decision Trees. *CoRR*, abs/2207.12213.

Audemard, G.; Koriche, F.; and Marquis, P. 2020. On Tractable XAI Queries based on Compiled Representations. In *KR*, 838–849.

Bailey, J.; and Stuckey, P. J. 2005. Discovery of Minimal Unsatisfiable Subsets of Constraints Using Hitting Set Dualization. In *PADL*, 174–186.

Belaid, M.; Bessiere, C.; and Lazaar, N. 2019. Constraint Programming for Association Rules. In Berger-Wolf, T. Y.; and Chawla, N. V., eds., *SIAM*, 127–135.

Belov, A.; Lynce, I.; and Marques-Silva, J. 2012. Towards efficient MUS extraction. *AI Commun.*, 25(2): 97–116.

Bendík, J.; Cerná, I.; and Benes, N. 2018. Recursive Online Enumeration of All Minimal Unsatisfiable Subsets. In *ATVA*, 143–159.

Biere, A.; Heule, M.; van Maaren, H.; and Walsh, T., eds. 2021. *Handbook of Satisfiability*. IOS Press. ISBN 978-1-64368-160-3.

Blanc, G.; Lange, J.; and Tan, L. 2021. Provably efficient, succinct, and precise explanations. In *NeurIPS*.

Boumazouza, R.; Alili, F. C.; Mazure, B.; and Tabia, K. 2021. ASTERYX: A model-Agnostic SaT-basEd appRoach for sYmbolic and score-based eXplanations. In *CIKM*, 120–129.

Camburu, O.; Giunchiglia, E.; Foerster, J.; Lukasiewicz, T.; and Blunsom, P. 2019. Can I Trust the Explainer? Verifying Post-hoc Explanatory Methods. *CoRR*, abs/1910.02065.

Chen, T.; and Guestrin, C. 2016. XGBoost: A Scalable Tree Boosting System. In *KDD*, 785–794.

Clark, P.; and Niblett, T. 1989. The CN2 Induction Algorithm. *Machine Learning*, 3: 261–283.

Darwiche, A.; and Hirth, A. 2020. On the Reasons Behind Decisions. In *ECAI*, 712–720.

Darwiche, A.; and Marquis, P. 2021. On Quantifying Literals in Boolean Logic and Its Applications to Explainable AI. *J. Artif. Intell. Res.*, 72: 285–328.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

Ferreira, J.; de Sousa Ribeiro, M.; Gonçalves, R.; and Leite, J. 2022. Looking Inside the Black-Box: Logic-based Explanations for Neural Networks. In *KR*, 432–442.

Friedman, J. H. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29(5): 1189–1232.

Gario, M.; and Micheli, A. 2015. PySMT: a Solver-Agnostic Library for Fast Prototyping of SMT-Based Algorithms. In *SMT Workshop*.

Gorji, N.; and Rubin, S. 2022. Sufficient Reasons for Classifier Decisions in the Presence of Domain Constraints. In *AAAI*, 5660–5667.

Hipp, J.; Güntzer, U.; and Nakhaeizadeh, G. 2000. Algorithms for Association Rule Mining - A General Survey and Comparison. *SIGKDD Explor.*, 2(1): 58–64.

Huang, X.; Izza, Y.; Ignatiev, A.; Cooper, M. C.; Asher, N.; and Marques-Silva, J. 2022. Tractable Explanations for d-DNNF Classifiers. In *AAAI*, 5719–5728.

Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized Neural Networks. In *NIPS*, 4107–4115.

Ignatiev, A. 2020. Towards Trustable Explainable AI. In *IJCAI*, 5154–5158.

Ignatiev, A.; Izza, Y.; Stuckey, P. J.; and Marques-Silva, J. 2022. Using MaxSAT for Efficient Explanations of Tree Ensembles. In *AAAI*, 3776–3785.

Ignatiev, A.; Lam, E.; Stuckey, P. J.; and Marques-Silva, J. 2021. A Scalable Two Stage Approach to Computing Optimal Decision Sets. In *AAAI*, 3806–3814.

Ignatiev, A.; and Marques-Silva, J. 2021. SAT-Based Rigorous Explanations for Decision Lists. In *SAT*, 251–269.

Ignatiev, A.; Morgado, A.; and Marques-Silva, J. 2018. PySAT: A Python Toolkit for Prototyping with SAT Oracles. In *SAT*, 428–437.

Ignatiev, A.; Morgado, A.; and Marques-Silva, J. 2019. RC2: an Efficient MaxSAT Solver. *J. Satisf. Boolean Model. Comput.*, 11(1): 53–64.

Ignatiev, A.; Narodytska, N.; Asher, N.; and Marques-Silva, J. 2020. From Contrastive to Abductive Explanations and Back Again. In *AI*IA*, 335–355.

Ignatiev, A.; Narodytska, N.; and Marques-Silva, J. 2019a. Abduction-Based Explanations for Machine Learning Models. In *AAAI*, 1511–1519.

Ignatiev, A.; Narodytska, N.; and Marques-Silva, J. 2019b. On Relating Explanations and Adversarial Examples. In *NeurIPS*, 15857–15867.

Ignatiev, A.; Narodytska, N.; and Marques-Silva, J. 2019c. On Validating, Repairing and Refining Heuristic ML Explanations. *CoRR*, abs/1907.02509.

Ignatiev, A.; Previti, A.; Liffiton, M. H.; and Marques-Silva, J. 2015. Smallest MUS Extraction with Minimal Hitting Set Dualization. In *CP*, 173–182.

Izza, Y.; Ignatiev, A.; and Marques-Silva, J. 2022. On Tackling Explanation Redundancy in Decision Trees. *J. Artif. Intell. Res.*, 75: 261–321.

Izza, Y.; Jabbour, S.; Raddaoui, B.; and Boudane, A. 2020. On the Enumeration of Association Rules: A Decomposition-based Approach. In Bessiere, C., ed., *IJCAI*, 1265–1271.

Izza, Y.; and Marques-Silva, J. 2021. On Explaining Random Forests with SAT. In *IJCAI*.

Kohavi, R. 1996. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. In *KDD*, 202–207.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature*, 521(7553): 436.

Liffiton, M. H.; Previti, A.; Malik, A.; and Marques-Silva, J. 2016. Fast, flexible MUS enumeration. *Constraints An Int. J.*, 21(2): 223–250.

Liffiton, M. H.; and Sakallah, K. A. 2008. Algorithms for Computing Minimal Unsatisfiable Subsets of Constraints. *J. Autom. Reasoning*, 40(1): 1–33.

Lipton, Z. C. 2018. The mythos of model interpretability. *Commun. ACM*, 61(10): 36–43.

Lundberg, S. M.; and Lee, S. 2017. A Unified Approach to Interpreting Model Predictions. In *NeurIPS*, 4765–4774.

Malfa, E. L.; Michelmore, R.; Zbrzezny, A. M.; Paoletti, N.; and Kwiatkowska, M. 2021. On Guaranteed Optimal Robust Explanations for NLP Models. In *IJCAI*, 2658–2665.

Marques-Silva, J.; Gerspacher, T.; Cooper, M. C.; Ignatiev, A.; and Narodytska, N. 2020. Explaining Naive Bayes and Other Linear Classifiers with Polynomial Time and Delay. In *NeurIPS*.

Marques-Silva, J.; Gerspacher, T.; Cooper, M. C.; Ignatiev, A.; and Narodytska, N. 2021. Explanations for Monotonic Classifiers. In *ICML*, 7469–7479.

Marques-Silva, J.; Heras, F.; Janota, M.; Previti, A.; and Belov, A. 2013. On Computing Minimal Correction Subsets. In *IJCAI*, 615–622.

Marques-Silva, J.; and Ignatiev, A. 2022. Delivering Trustworthy AI through Formal XAI. In *AAAI*, 3806–3814.

Mencia, C.; Previti, A.; and Marques-Silva, J. 2015. Literal-Based MCS Extraction. In *IJCAI*, 1973–1979.

Miller, T. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.*, 267: 1–38.

Narodytska, N.; Shrotri, A. A.; Meel, K. S.; Ignatiev, A.; and Marques-Silva, J. 2019. Assessing Heuristic Machine Learning Explanations with Model Counting. In *SAT*, 267–278.

Olson, R. S.; Cava, W. G. L.; Orzechowski, P.; Urbanowicz, R. J.; and Moore, J. H. 2017. PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData Min.*, 10(1): 36:1–36:13.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Köpf, A.; Yang, E. Z.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*, 8024–8035.

Reiter, R. 1987. A Theory of Diagnosis from First Principles. *Artif. Intell.*, 32(1): 57–95.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *KDD*, 1135–1144.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2018. Anchors: High-Precision Model-Agnostic Explanations. In *AAAI*, 1527–1535.

Rivest, R. L. 1987. Learning Decision Lists. *Mach. Learn.*, 2(3): 229–246.

Rudin, C. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.*, 1(5): 206–215.

Shih, A.; Choi, A.; and Darwiche, A. 2018. A Symbolic Approach to Explaining Bayesian Network Classifiers. In *IJCAI*, 5103–5111.

Slack, D.; Hilgard, S.; Jia, E.; Singh, S.; and Lakkaraju, H. 2020. Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In *AIES*, 180–186.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR (Poster)*.

Wäldchen, S.; MacDonald, J.; Hauch, S.; and Kutyniok, G. 2021. The Computational Complexity of Understanding Binary Classifier Decisions. *J. Artif. Intell. Res.*, 70: 351–387.

Yu, J.; Ignatiev, A.; Stuckey, P. J.; Narodytska, N.; and Marques-Silva, J. 2022. Eliminating The Impossible, Whatever Remains Must Be True. *CoRR*, abs/2206.09551.

Zaki, M. J.; Parthasarathy, S.; Ogihara, M.; and Li, W. 1997. New Algorithms for Fast Discovery of Association Rules. In *KDD*, 283–286.

Zhang, C.; and Zhang, S. 2002. *Association Rule Mining, Models and Algorithms*.