On Relating Explanations and Adversarial Examples

Alexey Ignatiev
Monash University, Australia
alexey.ignatiev@monash.edu

Nina Narodytska
VMWare Research, CA, USA
nnarodytska@vmware.com

Joao Marques-Silva
ANITI, Toulouse, France
joao.marques-silva@univ-toulouse.fr

Abstract

The importance of explanations (XP's) of machine learning (ML) model predictions and of adversarial examples (AE's) cannot be overstated, with both arguably being essential for the practical success of ML in different settings. There has been recent work on understanding and assessing the relationship between XP's and AE's. However, such work has been mostly experimental and a sound theoretical relationship has been elusive. This paper demonstrates that explanations and adversarial examples are related by a generalized form of hitting set duality, which extends earlier work on hitting set duality observed in model-based diagnosis and knowledge compilation. Furthermore, the paper proposes algorithms, which enable computing adversarial examples from explanations and vice-versa.

1 Introduction

Adversarial examples (AE's) [54] illustrate the brittleness of machine learning (ML) models, and have been the subject of growing interest in recent years. Explanations (XP's) of (black-box) ML models provide trust in ML models, and exemplify the increasing importance of eXplainable AI (XAI) [17, 12, 13]. Over the last few years, a number of works realized the existence of some connection between AE's and XP's [34, 55, 48, 56, 59, 41, 8]. However, past work has been experimental, and a deeper theoretical connection between AE's and XP's has been elusive. This paper demonstrates the existence of such a theoretical connection between AE's and XP's.

In this work we take a formal logic point of view on the analysis of ML models. Namely, we employ first order logic (FOL) as a framework to specify an ML model, define notions of an explanation and a counterexample to that explanation, which can be viewed as a generalization of an adversarial example. We then demonstrate that these notions possess well-known counterparts in the FOL terminology, like prime implicants and implicates, respectively. Such formalization allows us to obtain our main result that reveals a duality relation between explanations and counterexamples. Based on this connection, we show how explanations and counterexample interact. For example, explanations can be used to generate counterexamples. Dually, we can generate explanations given counterexamples. Furthermore, we also show how to compute adversarial examples from counterexamples. The ideas in the paper build on tightly related work in model-based diagnosis [45], namely the hitting set duality between diagnoses and conflicts, but also builds on related work in knowledge compilation, concretely in the use of prime implicants and implicates to compile knowledge [51, 36].

The paper is organized as follows. Section 2 introduces concepts used in the remainder of the paper. Section 3 investigates the connections between adversarial examples and explanations, and proposes algorithms for the enumeration of explanations and adversarial examples. Experimental

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

evidence demonstrating the relationship between explanations and adversarial examples is analyzed in Section 4. The paper concludes in Section 5.

2 Background

2.1 Preliminaries

We consider an ML model $\mathbb{M}$, represented by a finite set of first order logic (FOL) sentences $\mathcal{M}$. (Where viable, alternative representations for $\mathcal{M}$ can be considered, e.g. fragments of FOL, (mixed-)integer linear programming, constraint language(s), etc.) A set of features $\mathcal{F} = \{f_1, \ldots, f_L\}$ is assumed. Each feature f_i is categorical (or ordinal), with values taken from some set D_i. An *instance* is an assignment of values to features. The space of instances, also referred to as *feature* (or *instance*) *space*, is defined by $\mathbb{F} = D_1 \times D_2 \times \ldots \times D_L$. (Domains may or may not have an order relation. Also, for real-valued features, a suitable interval discretization can be considered.) A (feature) literal λ_i is of the form $(f_i = v_i)$, with $v_i \in D_i$. In what follows, a literal will be viewed as an atom, i.e. it can take value *true* or *false*. As a result, an instance can be viewed as a set of L literals, denoting the L distinct features, i.e. an instance contains a single occurrence of a literal defined on any given feature. A set of literals is consistent if there exists at most one literal associated with each feature. A consistent set of literals can be interpreted as a conjunction or as a disjunction of literals; this will be clear from the context. When interpreted as a conjunction, the set of literals denotes a *cube* in instance space, where the unspecified features can take any possible value of their domain. When interpreted as a disjunction, the set of literals denotes a *clause* in instance space. As before, the unspecified features can take any possible value of their domain.

The remainder of the paper assumes a classification problem with a set of classes $\mathbb{K} = \{\kappa_1, \ldots, \kappa_M\}$. A prediction $\pi \in \mathbb{K}$ is associated with each instance $\mathcal{I}$.

Given some target prediction $\pi \in \mathbb{K}$, one can devise representations for the formula $F_{\mathcal{M},\pi} \triangleq (\mathcal{M} \rightarrow \pi)$ [52, 23]. In particular, we will be interested in computing prime implicants and implicates of $F_{\mathcal{M},\pi}$, where a consistent set of feature literals τ is an implicant of $F_{\mathcal{M},\pi}$ if $\tau \vDash F_{\mathcal{M},\pi}$, and a consistent set of feature literals ν is a (negated) implicate of $F_{\mathcal{M},\pi}$ if $F_{\mathcal{M},\pi} \vDash \neg\nu$, or alternatively $(\nu \vDash \neg F_{\mathcal{M},\pi}) \equiv (\nu \vDash \vee_{\rho \neq \pi}(\mathcal{M} \rightarrow \rho))$. An implicant τ (implicate ν, resp.) is called *prime* if none of its proper subsets $\tau' \subsetneq \tau$ ($\nu' \subsetneq \nu$, resp.) is an implicant (implicate, resp.).

Throughout the paper, it will be convenient to use a more detailed notation, where ML models, prime implicants and prime implicates represent functions, respectively mapping $\mathbb{F}$ into $\mathbb{K}$ and $\{0, 1\}$. Concretely, the ML model $\mathbb{M}$ computes a function $\mathcal{M} : \mathbb{F} \rightarrow \mathbb{K}$ [1]. As a result, given some instance $X \in \mathbb{F}$ in feature space, $\mathcal{M}(X)$ denotes the prediction computed by the ML model. Furthermore, the notation $\tau \vDash F_{\mathcal{M},\pi}$ represents the following first order logic statement:

$$\forall (X \in \mathbb{F}).\tau(X) \rightarrow (\mathcal{M}(X) = \pi) \tag{1}$$

where τ is a Boolean function mapping $\mathbb{F}$ into $\{0, 1\}$, and $\mathcal{M}$ is a function mapping $\mathbb{F}$ into $\mathbb{K}$. Essentially, a prime implicant is viewed as a Boolean function taking value 1 for a *cube* (i.e. set of points) in feature space for which the prediction is π. Similarly, the notation $\nu \vDash \neg F_{\mathcal{M},\pi}$ represents the following first order logic statement:

$$\forall (X \in \mathbb{F}).\nu(X) \rightarrow (\vee_{\rho \neq \pi} \mathcal{M}(X) = \rho) \tag{2}$$

where ν is a Boolean function mapping $\mathbb{F}$ into $\{0, 1\}$.

Example 1. *The paper's running example is the restaurant example from Russell&Norvig's book [50, Fig. 18.3, page 700]. For this example, the set of features is:*

$$\{\mathsf{A(lternate)}, \mathsf{B(ar)}, \mathsf{W(eekend)}, \mathsf{H(ungry)}, \mathsf{Pa(trons)}, \mathsf{Pr(ice)}, \mathsf{Ra(in)}, \mathsf{Re(serv.)}, \mathsf{T(ype)}, \mathsf{E(stim.)}\}.$$

For instance, A, B, W, H, Ra, Re *are Boolean features taking* True *or* False *values.* T *is a categorical feature with four possible values* $\{\mathsf{Burger}, \mathsf{French}, \mathsf{Italian}, \mathsf{Thai}\}$. *The other domains are defined similarly. The dataset predicts whether the customer should wait or not in a given situation. So we have the target label* Wait *that takes* Yes *or* No *values. An example instance is:* $\{\mathsf{A}, \neg\mathsf{B}, \neg\mathsf{W}, \mathsf{H}, (\mathsf{Pa} = \mathsf{Some}), (\mathsf{Pr} = \$\$\$), \neg\mathsf{Ra}, \mathsf{Re}, (\mathsf{T} = \mathsf{French}), (\mathsf{E} = 0\text{–}10)\}$.

[1] Since formula $\mathcal{M}$ computes a function, the prediction for *any* instance is unique. It is straightfoward to devise ML models where $\mathcal{M}$ does not compute a function. Such cases are not considered in this paper.

Example 2. *Throughout the paper, the examples will consider decision sets [28]. (The selection of decision sets is motivated by simplicity. The actual experiments consider black-box models.) For the example dataset, a decision set obtained with an off-the-shelf tool is:*

IF	$(\mathsf{Pa} = \mathsf{Some}) \wedge \neg(\mathsf{E} = {>}60)$	**THEN**	$(\mathsf{Wait} = \mathsf{Yes})$	(R1)
IF	$\mathsf{W} \wedge \neg(\mathsf{Pr} = \$\$\$) \wedge \neg(\mathsf{E} = {>}60)$	**THEN**	$(\mathsf{Wait} = \mathsf{Yes})$	(R2)
IF	$\neg\mathsf{W} \wedge \neg(\mathsf{Pa} = \mathsf{Some})$	**THEN**	$(\mathsf{Wait} = \mathsf{No})$	(R3)
IF	$(\mathsf{E} = {>}60)$	**THEN**	$(\mathsf{Wait} = \mathsf{No})$	(R4)
IF	$\neg(\mathsf{Pa} = \mathsf{Some}) \wedge (\mathsf{Pr} = \$\$\$)$	**THEN**	$(\mathsf{Wait} = \mathsf{No})$	(R5)

2.2 Related Work

We overview two main research directions: (a) methods for generating adversarial attacks and (b) methods for producing explanations of ML model decisions. These two research directions have similarities, e.g. both types of methods make assumptions about transparency of the model, i.e. whether it is a white-box or a black-box, enforce different guarantees on the outcome (best effort vs guaranteed solution), etc. However, somewhat surprisingly, a formal connection between adversarial examples and explanations has not been proposed in the literature. Our work bridges this gap.

Adversarial attacks. Szegedy *et al.* demonstrated that ML models lack robustness: a small perturbation of an input may lead to a significant perturbation of the output of an ML model [54]. This vulnerability can be exploited to augment the original input with a crafted perturbation, invisible to a human but sufficient for the ML model to misclassify this input. A perturbation is required to be small w.r.t. a given metric, e.g. l_1, l_2, and l_∞ [54, 16, 6] norms. This influential work triggered several new research directions [7].

Depending on the threat model, adversarial attack generators can be broadly partitioned into black-box and white-box methods. Black-box methods assume that the attacker has no knowledge about the ML model. In this case, adversarial attacks are based on algorithms that recover the original model structure and transfer adversarial examples to the original model [42, 43, 27]. In contrast, white-box methods assume that the adversary has complete knowledge about the model, e.g. [54, 6, 37, 38]. In this work, we lean towards a black-box model, however we assume the existence of a logic-based oracle that can be queried about entailment relations between inputs and outputs. The majority of white-box methods to produce adversarial examples are heuristic and rely on gradient descent methods. Hence, they cannot guarantee that an adversarial example will be found even if it exists. For safety-critical applications such uncertainty might not be tolerable, therefore, a new trend is emerging focusing on methods with *provable guarantees* [25, 4, 33, 40, 10, 14, 26, 30]. For example, Katz *et al.* proposed the Reluplex system that finds an adversarial example if it exists in ReLU-based networks [25]. The main idea is to encode a network function as an SMT formula and to prove its properties (e.g. the absence of an adversarial perturbation in a given neighborhood) using an SMT and ILP hybrid. A formal approach was also applied to binarized neural networks [11, 40, 10, 26]. Finally, there is work on understanding adversarial examples. Xu *et al.* provide sensitivity analysis of pixel level perturbations and investigate the effect of these perturbations on internal layers of the network [58]. In [59], the authors produce structured attacks, where the attack mechanism achieves strong group sparsity leading to more interpretable examples.

Model explanations. Explainability of ML models depends on the type of the model that a user works with. There is a class of models considered to be interpretable by a human decision maker, like decision trees, lists or sets. When considering interpretable ML models, the goal is to compute models that provide *minimal* explanations associated with each prediction [28, 2, 39, 24, 49][2].

In case of non-interpretable models, like neural networks or ensembles of trees, there are two main options: (a) recompile (augment) the original model into (with) an explainable model [15, 57] or (b) extract an explanation from the model. The former approach might not be suitable if we want to provide explanations with guarantees w.r.t. the original model. The latter approach is currently the mainstream one. As in the case of adversarial attacks, the method designer needs to make an

[2]Explanations find a wide range of uses in AI and CS in general. In general, understanding the causes of inconsistency in over-constrained systems of constraints corresponds to explaining the reasons of inconsistency. In a similar fashion, diagnosis of failing systems can also be seen as explaining the reasons for system failure [45].

assumption on transparency of the model to the explainer. As above, white-box methods rely on computing gradients, e.g. saliency maps or integrated gradients [53]. However, these methods are mostly applicable to computer vision tasks. One influential line of research looks into explaining black-box models [19]. Explanations can be local [46, 35, 18] or global [47, 29], depending on whether they only apply to a local neighborhood of a target instance or not. While these methods can provide probabilistic guarantees, they do not provide worst-case guarantees on generated explanations. Moreover, there exist concerns regarding robustness of some of these methods [1].

Recently, two approaches were proposed to compute global explanations. The first method takes a compilation-based approach to computing global explanations [52]. If it is possible to compile an ML model to a suitable compilation structure, this method can extract all possible global explanations. However, the main drawback of this approach is exponential worst-case size of the compiled representation. In [23], the authors proposed new methods for computing explanations, by extracting prime implicants. This approach scales better compared to the compilation based approach and can generate a number of global explanations on demand. The current work is based on ideas from [23] to generate explanations and counterexamples.

There is a recent line of work on defending ML models against adversarial attacks based on interpretablity [34, 55]. For example, in [55] the authors identify neurons that correspond to human perceptible attributes and check whether these attributes are used in classification of the input. If so the input is non-adversarial and adversarial otherwise. Tomsett *et al.* [56] stated that adversarial examples and explanations are related notions. Namely, they argue that adversarial examples can improve ML interpretability and vice versa, e.g. neurons activations patterns are different for adversarial and original inputs which provides an insight about the network's internal representation. Finally, there is work on using advanced training procedures, like robust training, to improve the network interpretablity [48, 41, 8]. For example, in [48] the authors propose to regularize input gradients to improve robustness to transferred adversarial examples and quality of gradient-based explanations. Chalasani *et al.* proposed to employ adversarial training to learn logistic models with the feature-concentration property that are easier for the user to interpret [8].

3 Relating Explanations and Adversarial Examples

The goal of this section is to establish a tight connection between adversarial examples and explanations. To achieve this goal, we must first formalize the notion of (absolute) explanations and that of counterexample, and prove a (minimal) hitting set relationship between the two. Afterwards, we demonstrate how adversarial examples can be computed from explanations and vice-versa.

3.1 Explanations & Counterexamples

In this section, we introduce two new notions: *absolute explanations* and *counterexamples* over the input features. An absolute explanation for the prediction π is a generalization of commonly used local and global explanations [46, 47, 52, 23]. An absolute explanation is the strongest form of an explanation that does not depend on a concrete input instance and acts globally over the entire feature space. For any instance that *matches* an absolute explanation, the ML model prediction is guaranteed to be π. By matching, we mean that a set of features shared by the instance and the explanation have the same values. The second notion is a counterexample to the prediction π, which is a generalization of commonly used adversarial and some forms of universal adversarial examples [46, 37]. Intuitively, a counterexample is a set of input feature values that forces the ML model to output a prediction that is different from π. We are mostly interested in minimal such sets. Again, it is a strong notion as any instance that matches the counterexample must not be classified as π.

Given an ML model, represented by some logic encoding $\mathcal{M}$, and a prediction $\pi \in \mathbb{K}$, the following definitions are considered.

Definition 1 (Explanation)**.** *A(n absolute) explanation (XP) of a prediction π is a subset-minimal set*[3] *of literals $\mathcal{E}$, representing distinct features, such that $\mathcal{E} \models (\mathcal{M} \rightarrow \pi)$.*

[3]Given a set $\mathcal{R}$, subset-minimality of a set $\varphi \subseteq \mathcal{R}$ wrt. a predicate $\mathcal{P}$ over set $\mathcal{R}$ means that (a) $\mathcal{P}(\varphi)$ and (b) $\forall(\varphi' \subsetneq \varphi)\, \neg\mathcal{P}(\varphi')$.

Observe that explanations are often deemed as local [46, 47]. Alternatively, *global* explanations (although hold in the complete instance space) are relative to an instance $\mathcal{I}$, when $\mathcal{E} \subseteq \mathcal{I}$ [23]. Explanations in this paper are independent of a concrete instance, and so are referred to as *absolute*.

Definition 2 (Counterexample)**.** *A subset-minimal set $\mathcal{C}$ of literals is a* counterexample *(CEx) to a prediction π, if $\mathcal{C} \vDash (\mathcal{M} \rightarrow \rho)$, with $\rho \in \mathbb{K} \wedge \rho \neq \pi$.*

Clearly, an explanation $\mathcal{E}$ is a prime implicant of $F_{\mathcal{M},\pi}$ and a counterexample $\mathcal{C}$ is a (negated) prime implicate of $F_{\mathcal{M},\pi}$.

Example 3. *For the running example, due to* (R1)*, an explanation for the prediction* $(\mathsf{Wait} = \mathsf{Yes})$ *is:* $(\mathsf{Pa} = \mathsf{Some}) \wedge \neg(\mathsf{E} = >60)$*. Moreover, due to* (R5)*, a counterexample for the prediction* $(\mathsf{Wait} = \mathsf{Yes})$ *is:* $\neg(\mathsf{Pa} = \mathsf{Some}) \wedge (\mathsf{Pr} = \$\$\$)$.

Two literals are *inconsistent* if they represent the same feature but refer to different values. We say that a literal τ_i *breaks* a set of literals $\mathcal{S}$ (each denoting a different feature) if $\mathcal{S}$ contains a literal inconsistent with τ_i. Thus we can talk about breaking an explanation $\mathcal{E}$ or breaking a counterexample $\mathcal{C}$. Moreover, two sets of literals $\mathcal{S}_1$ and $\mathcal{S}_2$ break each other if they contain literals in the same feature referring to different values.

Example 4. *For the running example, the explanation corresponding to the set of literals* $\mathcal{S}_1 = \{(\mathsf{Pa} = \mathsf{Some}), \neg(\mathsf{E} = >60)\}$ *breaks the counterexample corresponding to the set of literals* $\mathcal{S}_2 = \{\neg(\mathsf{Pa} = \mathsf{Some}), (\mathsf{Pr} = \$\$\$)\}$ *and vice-versa, as* $(\mathsf{Pa} = \mathsf{Some})$ *and* $\neg(\mathsf{Pa} = \mathsf{Some})$ *are the inconsistent literals in this case.*

As hinted in the examples above, we can now state the paper's main result. We start with a general assumption.

Assumption 1. *The ML model $\mathbb{M}$ computes a function $\mathcal{M} : \mathbb{F} \rightarrow \mathbb{K}$.*

This assumption is essential to ensure that for any instance in feature space the prediction is unique.

Theorem 1. *Given an ML model $\mathbb{M}$, represented by some logic encoding $\mathcal{M}$, and a prediction π, every explanation $\mathcal{E}$ of π* breaks every *counterexample of π, and every counterexample $\mathcal{C}$ of π* breaks every *explanation of π.*

Proof. The proof of the theorem statement consists of two parts:

1. Every explanation $\mathcal{E}$ of π breaks every counterexample $\mathcal{C}$ of π.

$\forall(X \in \mathbb{F}).\mathcal{C}(X) \rightarrow (\vee_{\rho \neq \pi} \mathcal{M}(X) = \rho)$	Definition of some counterexample $\mathcal{C}$
$\forall(X \in \mathbb{F}).\neg(\vee_{\rho \neq \pi} \mathcal{M}(X) = \rho) \rightarrow \neg\mathcal{C}(X)$	Contrapositive
$\forall(X \in \mathbb{F}).(\mathcal{M}(X) = \pi) \rightarrow \neg\mathcal{C}(X)$	Negation given set of classes
$\forall(X \in \mathbb{F}).\mathcal{E}(X) \vDash (\mathcal{M}(X) = \pi)$	Definition of some explanation $\mathcal{E}$
$\forall(X \in \mathbb{F}).\mathcal{E}(X) \vDash \neg\mathcal{C}(X)$	Explanation breaks counterexample

2. Every counterexample $\mathcal{C}$ of π breaks every explanation $\mathcal{E}$ of π.

$\forall(X \in \mathbb{F}).\mathcal{E}(X) \rightarrow (\mathcal{M}(X) = \pi)$	Definition of some explanation $\mathcal{E}$
$\forall(X \in \mathbb{F}).\neg(\mathcal{M}(X) = \pi) \rightarrow \neg\mathcal{E}(X)$	Contrapositive
$\forall(X \in \mathbb{F}). \vee_{\rho \neq \pi} (\mathcal{M}(X) = \rho) \rightarrow \neg\mathcal{E}(X)$	Negation given set of classes
$\forall(X \in \mathbb{F}).\mathcal{C}(X) \vDash \vee_{\rho \neq \pi}(\mathcal{M}(X) = \rho)$	Definition of some counterexample $\mathcal{C}$
$\forall(X \in \mathbb{F}).\mathcal{C}(X) \vDash \neg\mathcal{E}(X)$	Counterexample breaks explanation

□

As argued below, by listing all minimal counterexamples, we can extract all minimal explanations, and vice-versa. Furthermore, one can readily conclude that Theorem 1 generalizes the hitting set relationship between diagnoses and conflicts first investigated by Reiter in the 80s [45], and since then studied in different settings [5, 3, 32].

Example 5 (Duality)**.** *For the running example, let the prediction again be* $(\mathsf{Wait} = \mathsf{Yes})$ *and the decision set proposed in Example 2. For this prediction and given the ML model, there are two global explanations:*

1. $(\mathsf{Pa} = \mathsf{Some}) \wedge \neg(\mathsf{E} = >60)$*; and*
2. $\mathsf{W} \wedge \neg(\mathsf{Pr} = \$\$\$) \wedge \neg(\mathsf{E} = >60)$.

This means that as long as any of these two conjunctions of literals holds, then the prediction will be $(\mathsf{Wait} = \mathsf{Yes})$*. Moreover, there are three counterexamples (i.e. explanations for not predicting* $(\mathsf{Wait} = \mathsf{Yes})$ *(which for this example corresponds to predicting* $(\mathsf{Wait} = \mathsf{No})$*):*

1. $\neg\mathsf{W} \wedge \neg(\mathsf{Pa} = \mathsf{Some})$*;*
2. $(\mathsf{E} = >60)$*; and*
3. $\neg(\mathsf{Pa} = \mathsf{Some}) \wedge (\mathsf{Pr} = \$\$\$)$.

This means that as long as any of these three conjunctions of literals holds, then the prediction will **not** *be* $(\mathsf{Wait} = \mathsf{Yes})$. *It can be readily concluded that the XP's (minimally) break the CEx's and vice-versa.*

Remark 1. *As hinted in Example 5, and building on earlier work on model-based diagnosis and enumeration of prime implicants and implicates [45, 51, 44], it is straightforward to compute counterexamples from explanations and vice-versa:*

1. *If we have the set of explanations for a prediction, then we can compute the set of counterexamples as the* consistent *minimal breaks of the set of explanations.*
2. *Similarly, if we have the set of counterexamples, then we can compute the set of explanations as the* consistent *minimal breaks of the set of counterexamples.*

Remark 2. *The assumptions for relating explanations with counterexamples are fairly general. For example, features do not need to be ordinal. Clearly, adversarial examples expect features to be ordinal. This is covered in the next section.*

3.2 Relationship with Adversarial Examples

The previous section showed that each explanation breaks every counterexample, and that each counterexample breaks every explanation. This holds for any machine learning model for which the logic representation of the model computes a function mapping feature space $\mathbb{F}$ into $\mathbb{K}$. Adversarial examples were introduced in earlier work [54] *denoting small changes to the features w.r.t. to a given distance measure that results in the prediction error*. In this paper, we use the following definition of adversarial example. Given an instance $\mathcal{I}$ in feature space, corresponding to prediction π, our goal is to find another instance, $\mathcal{I}_{\mathsf{ae}}$, corresponding to a different prediction, and which is *closest* to the original instance. Formally,

$$\begin{array}{ll} \min & \mathrm{Dist}(\mathcal{I}_{\mathsf{ae}}, \mathcal{I}) \\ \mathrm{st} & \mathcal{I}_{\mathsf{ae}} \models \vee_{\rho \neq \pi} \rho \end{array} \tag{3}$$

Clearly, adversarial examples assume that features are ordinal, enabling the notion of distance to be well-defined. (For real-valued features, we assume an often-used discretization of the input.) We can now relate adversarial examples with counterexamples and with explanations.

Theorem 2 (From XP's to AE's)**.** *Given an ML model $\mathbb{M}$, represented with a logic representation $\mathcal{M}$, a prediction π, with set of explanations $\mathbb{E}$ and set of counterexamples $\mathbb{C}$, and an instance $\mathcal{I}$ taken from feature space, let $\mathcal{C}_{\mathsf{ae}}$ denote the counterexample with minimum distance to $\mathcal{I}$. Then $\mathcal{I}_{\mathsf{ae}}$ corresponds to $\mathcal{C}_{\mathsf{ae}}$ by setting the unspecified feature values to the values in $\mathcal{I}$.*

Proof (Sketch). If we know the set of explanations, then we can compute the set of counterexamples (see Remark 1).
Given the set of counterexamples, we can compute one that minimizes some measure of distance to the instance $\mathcal{I}$.
A counterexample represents a cube in feature space; we just need to pick the point closest to $\mathcal{I}$. This can be achieved by fixing the free features. Observe that $\mathcal{I}_{\mathsf{ae}}$ is the complete assignment obtained from the partial assignment $\mathcal{C}_{\mathsf{ae}}$ by setting the missing coordinates to the feature values specified in the given instance $\mathcal{I}$. □

3.3 Exploiting Duality

The duality between absolute explanations $\mathcal{E} \in \mathbb{E}$ and counterexamples $\mathcal{C} \in \mathbb{C}$ for a prediction π made by model $\mathbb{M}$ (represented as formula $\mathcal{M}$) can be exploited directly to compute either $\mathbb{E}$, or $\mathbb{C}$, or both. This can be done following the ideas of *prime compilation* of Boolean formulas [44]. Alternatively, the classifier can be compiled into a succinct logical representation, e.g. binary decision diagram (BDD) [52], which allows for efficient enumeration of prime implicants and implicates.

Algorithm 1 shows a Pythonic-style algorithm to compute the complete set $\mathbb{E}$. It computes the set $\mathbb{E}$ of all explanations and a subset of all the counterexamples $\mathbb{C}$. The algorithm utilizes the hitting set duality and represents the *implicit hitting set enumeration* paradigm [9]. It can be seen as a loop, each iteration of which computes a smallest size hitting set $\mathcal{E}$ of the set $\mathbb{C}$ of counterexamples (see

Algorithm 1: Duality-based computation of all absolute explanations

Input: formula $\mathcal{M}$ and prediction π
Output: set $\mathbb{E}$ of all absolute explanations of prediction π

1 $(\mathbb{C}, \mathbb{E}, \mathcal{E}) \leftarrow (\emptyset, \emptyset, \emptyset)$
2 **do:**
3   **if** $\mathcal{E} \models (\mathcal{M} \rightarrow \pi)$ **:**
4         $\mathbb{E} \leftarrow \mathbb{E} \cup \{\mathcal{E}\}$ `# ` $\mathcal{E}$ `is an explanation; save it`
5   **else:**
6         $(\mathcal{C}, \rho) \leftarrow$ `ExtractInstance()` `# get an instance` $\mathcal{C}$ `with a prediction` ρ, $\rho \neq \pi$
7   **for** $l \in \mathcal{C}$ **:**
8   **if** $(\mathcal{C} \setminus \{l\}) \models (\mathcal{M} \rightarrow \rho)$ **:**
9                 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{l\}$
10         $\mathbb{C} \leftarrow \mathbb{C} \cup \{\mathcal{C}\}$ `# update` $\mathbb{C}$ `with a new counterexample` $\mathcal{C}$
11     $\mathcal{E} \leftarrow$ `MinimumHS`$(\mathbb{C})$ `# get a new hitting set of` $\mathbb{C}$
12 **while** $\mathcal{E} \neq \emptyset$
13 **return** $\mathbb{E}$

line 11) and checks whether or not $\mathcal{E}$ is an explanation for prediction π (line 3). (This check can be done by calling an oracle testing unsatisfiability of formula $\mathcal{E} \wedge \mathcal{M} \wedge \neg\pi$.) If it is, $\mathcal{E}$ is added to $\mathbb{E}$. Otherwise, i.e. if $\mathcal{E} \wedge \mathcal{M} \wedge \neg\pi$ is satisfiable, a satisfying assignment exists defining an instance $\mathcal{C}$ that is classified by $\mathcal{M}$ as some $\rho \in \mathbb{K}$ s.t. $\rho \neq \pi$. Such satisfying assignment is typically easy to obtain from the oracle (line 6). Instance $\mathcal{C}$ is then reduced to a counterexample by removing all *redundant*, i.e. unnecessary, literals (see the loop line 7–line 9 and concretely the oracle call in line 8). The new counterexample is added to the set $\mathbb{C}$ (see line 10) and a new hitting set $\mathcal{E}$ is obtained (line 11). The algorithm proceeds until there is no more hitting set $\mathcal{E}$ of $\mathbb{C}$. Observe that *initially* $\mathcal{E}$ is empty and, thus, the first iteration of Algorithm 1 *always* results in a new counterexample $\mathcal{C}$ being computed and added to $\mathbb{C}$. Note that although Algorithm 1 targets enumerating all explanations $\mathbb{E}$, it can also be applied for computing the set $\mathbb{C}$ of all counterexamples, with minimal modifications of the formulas in line 3 and line 8.

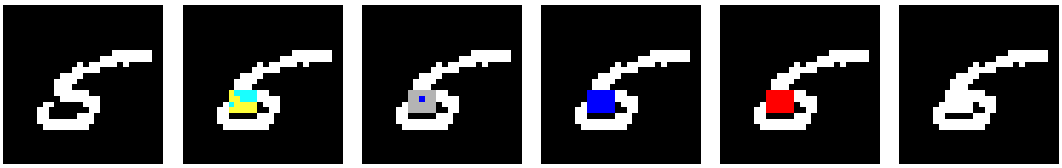
Remark 3. *Although the goal of Algorithm 1 is to illustrate a way to exploit the duality, its practical efficiency may not be ideal in some specific settings. The algorithm relies on the oracle calls in line 3 and line 8, which are NP-hard. Also, extracting a smallest size hitting set is NP-hard as well and can be done with the use of modern optimization procedures, e.g. mixed integer programming (MILP) [20] or maximum satisfiability (MaxSAT) [31]. Furthermore, in the worst case, the algorithm could end up enumerating both all explanations and all counterexamples and there might be an exponential number of them. However, in other settings, this worst-case scenario is not often observed in practice.*

4 Experimental Evidence

The section practically illustrates the described duality between the concepts of absolute explanation and counterexample for a given model prediction. To do this, the following experiment was performed on a Macbook Pro with an Intel Core i5 2.3GHz CPU and 16GB of memory. The experiment targets the well-known and widely used MNIST digits database[4] as it enables a *visual demonstration* of the discovered duality relationship. As a classifier model, we consider neural networks (NNs) with *rectified linear unit* (ReLU) non-linear activation operators and the known encoding of ReLU-based NNs into MILP [14]. The developed Python-based prototype[5] follows the prime compilation approach of Algorithm 1 and uses CPLEX 12.8.0 [20] as an MILP oracle, which is invoked at each iteration of the algorithm. The implementation of minimum hitting set enumeration of Algorithm 1 is based on an award-winning maximum satisfiability solver RC2[6] [22] written on top of the PySAT toolkit [21].

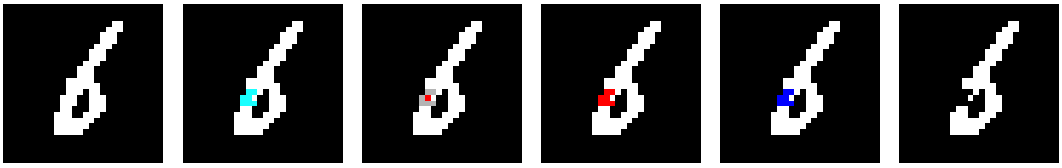
For the sake of simplicity, the networks used are trained to distinguish two digits, e.g. 5 and 6 (because of their visual resemblance). Also, due to a significant number of explanations and counterexamples,

[4] http://yann.lecun.com/exdb/mnist/
[5] https://github.com/alexeyignatiev/xpce-duality/
[6] https://maxsat-evaluations.github.io/



(a) digit “5” (b) patch area (c) an XP (d) all XP’s (e) a CE (f) an AE (“6”)

Figure 1: An example of digit five.



(a) digit “6” (b) patch area (c) an XP (d) all XP’s (e) a CE (f) an AE (“5”)

Figure 2: An example of digit six.

the following is assumed: (1) only pixels from a predefined patch area can participate in an explanation/counterexample with the other pixels being fixed; (2) the images were binarized, i.e. every pixel can be either black or white. However, note that the duality holds for the most general case with assumptions (1) and (2) being disabled.

Figure 1a and Figure 2a show two concrete examples of digits 5 and 6. The patch areas for these images are highlighted in Figure 1b and Figure 2b. The patches contains 20 and 7 pixels, respectively. These patch areas are selected intentionally as their pixels are supposed to be crucial for the prediction being “5” or “6”. Enumerating all explanations and counterexamples for these images with the given patches result in 20 (7, resp.) unit-size explanations for digit five (six, resp.). An example of one concrete explanation for these images is shown in Figure 1c and Figure 2c, respectively. Recall that the images are binarized; here, the corresponding pixel is blue (red, resp.) if the explanation sets it black (white, resp.) while the other (gray) pixels of the patch may have *any* color and the prediction will still remain as long as the pixels outside of the patch area are fixed. The unions of all explanations are shown in Figure 1d and Figure 2d, respectively. Also, for both images there is a unique counterexample. These are depicted in Figure 1e and Figure 2e. Observe that “polarities” of the pixels in explanations and counterexamples are opposite to each other. This clearly exhibits the described duality between the concepts of absolute explanations and counterexamples.

Note that the only counterexample for prediction “5” sets all pixels white. Such an image is shown in Figure 1f and represents an adversarial example for Figure 1a, i.e. it is classified as “6”. A similar observation can be made with respect to digit six. However, in this case the only counterexample sets all patch pixels black. Thus, an adversarial example for digit six is shown in Figure 2f and it is classified as “5”.

5 Conclusions

Adversarial examples and explanations of ML models are arguably two of the most significant areas of research in ML. This paper shows a tight relationship between the two. Concretely, the paper proposes the dual concept of counterexample, the notion of breaking an explanation or a counterexample, and shows that each explanation must break every counterexample and vice-versa. This property is tightly related with the concept of hitting set duality between diagnoses and conflicts in model-based diagnosis [45], but also with computation of prime implicants and implicates of Boolean functions [51]. The paper also overviews algorithms for computing explanations from counterexamples and vice-versa. Furthermore, the paper shows how adversarial examples can be computed given a reference instance in feature space and counterexample that minimizes the distance to the instance. The experimental evidence illustrates the applicability of the duality relationship between explanations and counterexamples (and adversarial examples). Future work will investigate extensions to the work to target problems of larger scale.

References

[1] D. Alvarez-Melis and T. S. Jaakkola. On the robustness of interpretability methods. *CoRR*, abs/1806.08049, 2018.

[2] E. Angelino, N. Larus-Stone, D. Alabi, M. Seltzer, and C. Rudin. Learning certifiably optimal rule lists. In *KDD*, pages 35–44, 2017.

[3] J. Bailey and P. J. Stuckey. Discovery of minimal unsatisfiable subsets of constraints using hitting set dualization. In *PADL*, pages 174–186, 2005.

[4] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. V. Nori, and A. Criminisi. Measuring neural net robustness with constraints. In *NIPS*, pages 2613–2621, 2016.

[5] E. Birnbaum and E. L. Lozinskii. Consistent subsets of inconsistent systems: structure and behaviour. *J. Exp. Theor. Artif. Intell.*, 15(1):25–46, 2003.

[6] N. Carlini and D. A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57. IEEE Computer Society, 2017.

[7] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay. Adversarial attacks and defences: A survey. *CoRR*, abs/1810.00069, 2018.

[8] P. Chalasani, S. Jha, A. Sadagopan, and X. Wu. Adversarial learning and explainability in structured datasets. *CoRR*, abs/1810.06583, 2018.

[9] K. Chandrasekaran, R. M. Karp, E. Moreno-Centeno, and S. Vempala. Algorithms for implicit hitting set problems. In *SODA*, pages 614–629, 2011.

[10] C. Cheng, G. Nührenberg, and H. Ruess. Verification of binarized neural networks. *CoRR*, abs/1710.03107, 2017.

[11] M. Courbariaux and Y. Bengio. Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1. *CoRR*, abs/1602.02830, 2016.

[12] DARPA. DARPA explainable Artificial Intelligence (XAI) program, 2016.

[13] EU Data Protection Regulation. Regulation (EU) 2016/679 of the European Parliament and of the Council, 2016.

[14] M. Fischetti and J. Jo. Deep neural networks and mixed integer linear optimization. *Constraints*, 23(3):296–309, 2018.

[15] N. Frosst and G. E. Hinton. Distilling a neural network into a soft decision tree. In *CExAIIA*, 2017.

[16] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[17] B. Goodman and S. R. Flaxman. European Union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, 38(3):50–57, 2017.

[18] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti. Local rule-based explanations of black box decision systems. *CoRR*, abs/1805.10820, 2018.

[19] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51(5):93:1–93:42, 2019.

[20] IBM ILOG: CPLEX optimizer 12.8.0. http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer, 2018.

[21] A. Ignatiev, A. Morgado, and J. Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *SAT*, pages 428–437, 2018.

[22] A. Ignatiev, A. Morgado, and J. Marques-Silva. RC2: An efficient MaxSAT solver. *Journal on Satisfiability, Boolean Modeling and Computation*, 11:53–64, 2019.

[23] A. Ignatiev, N. Narodytska, and J. Marques-Silva. Abduction-based explanations for machine learning models. In *AAAI*, 2019.

[24] A. Ignatiev, F. Pereira, N. Narodytska, and J. Marques-Silva. A SAT-based approach to learn explainable decision sets. In *IJCAR*, pages 627–645, 2018.

[25] G. Katz, C. W. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *CAV (1)*, pages 97–117, 2017.

[26] E. B. Khalil, A. Gupta, and B. Dilkina. Combinatorial attacks on binarized neural networks. *CoRR*, abs/1810.03538, 2018.

[27] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017.

[28] H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *KDD*, pages 1675–1684, 2016.

[29] H. Lakkaraju, E. Kamar, S. Caruana, and J. Leskovec. Faithful and customizable explanations of black box models. In *AIES*, 2019.

[30] F. Leofante, N. Narodytska, L. Pulina, and A. Tacchella. Automated verification of neural networks: Advances, challenges and perspectives. *CoRR*, abs/1805.09938, 2018.

[31] C. M. Li and F. Manyà. MaxSAT, hard and soft constraints. In *Handbook of Satisfiability*, pages 613–631. 2009.

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

[33] C. Liu, T. Arnon, C. Lazarus, C. Barrett, and M. J. Kochenderfer. Algorithms for verifying deep neural networks. *CoRR*, abs/1903.06758, 2019.

[34] N. Liu, H. Yang, and X. Hu. Adversarial detection with model interpretation. In *KDD*, pages 1803–1811, 2018.

[35] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

[36] P. Marquis. Consequence finding algorithms. In *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 41–145. 2000.

[37] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. *CoRR*, abs/1610.08401, 2016.

[38] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, P. Frossard, and S. Soatto. Robustness of classifiers to universal perturbations: A geometric perspective. In *ICLR*. OpenReview.net, 2018.

[39] N. Narodytska, A. Ignatiev, F. Pereira, and J. Marques-Silva. Learning optimal decision trees with SAT. In *IJCAI*, pages 1362–1368, 2018.

[40] N. Narodytska, S. P. Kasiviswanathan, L. Ryzhyk, M. Sagiv, and T. Walsh. Verifying properties of binarized deep neural networks. In *AAAI*, pages 6615–6624. AAAI Press, 2018.

[41] P. Panda and K. Roy. Explainable learning: Implicit generative modelling during training for adversarial robustness. *CoRR*, abs/1807.02188, 2018.

[42] N. Papernot, P. D. McDaniel, and I. J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.

[43] N. Papernot, P. D. McDaniel, I. J. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In R. Karri, O. Sinanoglu, A. Sadeghi, and X. Yi, editors, *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pages 506–519. ACM, 2017.

[44] A. Previti, A. Ignatiev, A. Morgado, and J. Marques-Silva. Prime compilation of non-clausal formulae. In *IJCAI*, pages 1980–1988. AAAI Press, 2015.

[45] R. Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32(1):57–95, 1987.

[46] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, editors, *KDD*, pages 1135–1144. ACM, 2016.

[47] M. T. Ribeiro, S. Singh, and C. Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI*, pages 1527–1535. AAAI Press, 2018.

[48] A. S. Ross and F. Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI*, pages 1660–1669, 2018.

[49] C. Rudin. Please stop explaining black box models for high stakes decisions. *CoRR*, abs/1811.10154, 2018.

[50] S. J. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach 3ed*. Prentice Hall, 2010.

[51] R. Rymon. An SE-tree-based prime implicant generation algorithm. *Ann. Math. Artif. Intell.*, 11(1-4):351–366, 1994.

[52] A. Shih, A. Choi, and A. Darwiche. A symbolic approach to explaining bayesian network classifiers. In *IJCAI*, pages 5103–5111, 2018.

[53] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR, 2017.

[54] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

[55] G. Tao, S. Ma, Y. Liu, and X. Zhang. Attacks meet interpretability: Attribute-steered detection of adversarial samples. In *NeurIPS*, pages 7728–7739, 2018.

[56] R. Tomsett, A. Widdicombe, T. Xing, S. Chakraborty, S. Julier, P. Gurram, R. M. Rao, and M. B. Srivastava. Why the failure? how adversarial examples can provide insights for interpretable machine learning. In *FUSION*, pages 838–845, 2018.

[57] T. Wang and Q. Lin. Hybrid Predictive Model: When an Interpretable Model Collaborates with a Black-box Model. *arXiv e-prints*, page arXiv:1905.04241, May 2019.

[58] K. Xu, S. Liu, G. Zhang, M. Sun, P. Z. and/ Quanfu Fan, C. Gan, and X. Lin. Interpreting adversarial examples by activation promotion and suppression. *CoRR*, abs/1904.02057, 2019.

[59] K. Xu, S. Liu, P. Zhao, P. Chen, H. Zhang, D. Erdogmus, Y. Wang, and X. Lin. Structured adversarial attack: Towards general implementation and better interpretability. *CoRR*, abs/1808.01664, 2018.