

Space Explanations of Neural Network Classification

Faezeh Labbaf¹ , Tomáš Kolárik¹ , Martin Blichář^{1,4} ,
 Grigory Fedyukovich^{1,3} , Michael Wand² , Natasha Sharygina¹ 

¹ University of Lugano (USI), Lugano, Switzerland

² SUPSI, IDSIA, Lugano, Switzerland

³ Florida State University, United States of America

⁴ Ethereum Foundation

Abstract

We present a novel logic-based concept called *Space Explanations* for classifying neural networks that gives provable guarantees of the behavior of the network in continuous areas of the input feature space. To automatically generate space explanations, we leverage a range of flexible Craig interpolation algorithms and unsatisfiable core generation. Based on real-life case studies, ranging from small to medium to large size, we demonstrate that the generated explanations are more meaningful than those computed by state-of-the-art.

1 Introduction

Explainability of decision-making AI systems (XAI), and specifically neural networks (NNs), is a key requirement for deploying AI in sensitive areas [18]. A recent trend in explaining NNs is based on formal methods and logic, providing explanations for the decisions of machine learning systems [24, 31, 32, 41, 42, 44] accompanied by provable guarantees regarding their correctness. Yet, rigorous exploration of the *continuous* feature space requires to estimate *decision boundaries* with complex shapes. This, however, remains a challenge because existing explanations [24, 31, 32, 41, 42, 44] constrain only individual features and hence fail capturing *relationships* among the features that are essential to understand the reasons behind the multi-parametrized classification process.

We address the need to provide interpretations of NN systems that are as meaningful as possible using a novel concept of *Space Explanations*, delivered by a flexible symbolic reasoning framework where Craig interpolation [12] is at the heart of the machinery. When starting from a sample point, the explanation

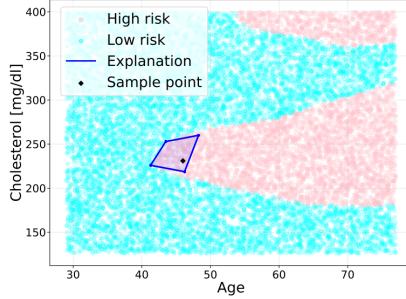


Figure 1: Example of close approximations of non-trivial decision boundaries using meaningful explanations of the classification: a NN classifier of heart attack risk [25].

computation is quick, and yields generality and soundness of approximation. Interpolation extracts information from the proof of unsatisfiability which captures a reason why the given input cannot change classification. We showcase explanations for trained neural networks substantially more expressive and meaningful compared to prior work in formal XAI. For example, the classification of the heart attack risk is determined (among others) based on a non-trivial relation of age and cholesterol level of a patient, as illustrated in Figure 1. Our prototype implementation of the framework, $S_{\text{P}}\text{EX}_{\text{PL}}\text{AI}_{\text{N}}$, can automatically compute an explanation that closely approximates the shape of the decision boundary. Furthermore, such non-restricted explanations often cover many sample points at once.

This case study paper is an applied study using existing techniques (mainly Craig interpolation) in a new practical context (XAI). We share our lessons learned after experimenting with various real-life benchmarks, evaluate the quality of explanations generated using different interpolation techniques and further reduced using unsatisfiable cores. We demonstrate the benefits in carefully selected experiments with respect to the state-of-the-art to see how the techniques apply and how different strategies can be combined. For all kinds of benchmarks, no matter the size and the domain, the resulting explanations are more meaningful for the user. Moreover, computing interpolation-based explanations scales better with the size of the input than existing methods.

Related Work

Most of the classic approaches to XAI either perform analysis at the *unit (neuron) level* [17, 45], which however fails to capture global properties or interdependencies between units; are *gradient-based* methods [4, 43], which describe the network behavior around a particular sample; or *inversion* methods [30], which provide global views, but are still only approximating. There is also a range of *model-agnostic* methods [5, 29, 39], including SHAP [27] and LIME [38], which can be applied to any classifier (not only NNs), but often yield logically inconsistent explanations [32]. All [4, 5, 17, 27, 29, 30, 38, 39, 43, 45] rely on ap-

proximations or are *sample-based*, that is, the behavior outside the space of the underlying dataset or samples generated from the model remains unknown.

We focus on formal approaches to XAI that offer *strict guarantees*, exploiting computational engines like MARABOU [26] or SMT solvers [6, 10, 11, 16, 22, 34]. State-of-the-art methods in this area address only special cases of our concept: constraints over individual features, leading to interval-based explanations [24] that cannot reflect feature relationships and hence approximate complex decision boundaries such as in Figure 1. Section 4 demonstrates generalizations of the existing explanations, resulting in strictly larger spaces. Early methods in this domain aimed at so-called abductive explanations [23, 31, 32, 41, 42, 44] that are primarily sample-based, resulting in zero-volume spaces. The term *abduction* [1, 14, 36, 37] is misused, since unlike abductive explanations, it is not restricted to samples. All the approaches [23, 24, 31, 32, 41, 42, 44] also heavily depend on selecting a specific order of the features in which the constraints are relaxed. Furthermore, they might build an exponential number of verification queries to an off-the-shelf NN verification engine. In contrast, our method directly uses one satisfiability query to an interpolating SMT solver to generate an interpolant.

Acknowledgement

A similar version of this paper has been accepted to CAV 2025 [28].

2 Background

A classification problem [20] is concerned with mapping input data into a pre-defined set of classes $\mathcal{K} = \{c_1, \dots, c_K\}$. Given a set of *features* $\mathcal{F} = \{1, \dots, m\}$, each $i \in \mathcal{F}$ takes values from a discrete or continuous *domain* \mathcal{D}_i . The *feature space* is defined as $\mathbb{F} = \mathcal{D}_1 \times \mathcal{D}_2 \times \dots \times \mathcal{D}_m$. By $\mathbf{x} = (x_1, \dots, x_m)$ we denote a point in \mathbb{F} , where each x_i is a *feature variable* (if clear from context, called features) taking values from \mathcal{D}_i . A *sample point* $\mathbf{s} = (s_1, \dots, s_m) \in \mathbb{F}$ contains constants representing concretes value from $\mathcal{D}_i, \dots, \mathcal{D}_m$. A *classifier* $\mathcal{M} = (\mathcal{F}, \mathcal{D}, \mathbb{F}, \mathcal{K}, \kappa)$ contains a *classification function* $\kappa : \mathbb{F} \rightarrow \mathcal{K}$. An *instance* is a pair (\mathbf{s}, c) , where $\mathbf{s} \in \mathbb{F}$ and $c \in \mathcal{K}$ is a prediction $c = \kappa(\mathbf{s})$. Given a classifier $\mathcal{M} = (\mathcal{F}, \mathcal{D}, \mathbb{F}, \mathcal{K}, \kappa)$, the *class space* $\mathbb{F}_c \subseteq \mathbb{F}$ of $c \in \mathcal{K}$ is $\{\mathbf{x} \in \mathbb{F} \mid \kappa(\mathbf{x}) = c\}$. A *classification rule* for c is a formula φ_c such that $\forall \mathbf{x} \in \mathbb{F}. \varphi_c(\mathbf{x}) \iff \kappa(\mathbf{x}) = c$. If κ is defined on the whole feature space \mathbb{F} then it can be partitioned as $\mathbb{F} = \bigcup_{c \in \mathcal{K}} \mathbb{F}_c$, and for all $a, b \in \mathcal{K}$, if $a \neq b$ then $\mathbb{F}_a \cap \mathbb{F}_b = \emptyset$ (because κ must not be ambiguous). A *class boundary* of c is the topological boundary of \mathbb{F}_c . The *decision boundary* is the union of class boundaries of all classes $c \in \mathcal{K}$ in a classifier $(\mathcal{F}, \mathcal{D}, \mathbb{F}, \mathcal{K}, \kappa)$.

3 Space Explanations

Explanations produced by existing tools do not capture relations between the input features, and consequently, they cannot approximate the decision boundary.

To address it, this paper presents a *logic-based* approach to compute explanations with the flexibility of representing different shapes of the feature space.

Definition 1 (Space Explanation, Impact Space). *Given a classifier \mathcal{M} computing a classification function κ from feature space \mathbb{F} , and a class c , a space explanation of c is a logic formula φ such that $\forall \mathbf{x} \in \mathbb{F}. \varphi(\mathbf{x}) \implies \kappa(\mathbf{x}) = c$. The impact space of φ , $\mathbb{F}_\varphi \subseteq \mathbb{F}_c$, is a set $\{\mathbf{x} \in \mathbb{F} | \varphi(\mathbf{x})\}$.*

Space explanations represent sufficient conditions of classification to class c . Since $\varphi \implies \varphi_c$, classification rules are also space explanations. Hence, we have a general concept of explanations with the following benefits:

1. The *shape* of the explanations is *not restricted*, hence it is possible to approximate arbitrarily complex class spaces and decision boundaries.
2. It is possible to *capture relationships* among features and truly *explain* the *reasons* behind the classification.
3. Space explanations can be merged or intersected with each other.

In Figure 1, for example, items 1 and 2 addressed by accurate approximation of the decision boundary using an explanation in the shape of a convex polygon.

We assume that for all classes, at least one sample point is classified into the class, meaning that none of the classes are redundant. Given a formula $\psi := \psi_{\mathcal{M}} \wedge \psi_{\mathcal{D}} \wedge \neg\psi_c$ where: $\psi_{\mathcal{M}}$ encodes the neural network \mathcal{M} , $\psi_{\mathcal{D}}$ encodes the domains \mathcal{D} of the feature space, i.e. $\mathbf{x} \in \mathbb{F}$, $\neg\psi_c$ encodes the constraint that the outcome of the classification is *not* class c , it is *satisfiable* with no additional restrictions. Now suppose a sample point \mathbf{s} classified as class c . Using the encoding $\varphi_{\mathbf{s}} := \bigwedge_{i \in \mathcal{F}} x_i = s_i$, formula $\varphi_{\mathbf{s}} \wedge \psi$ is in turn *unsatisfiable*.

This principle can be generalized by exploiting the fact that a space explanation φ of class c *guarantees* classification to the class for all points covered by the explanation, so formula $\varphi \wedge \psi$ is also unsatisfiable. Such a formula enables the use of Craig interpolation¹: interpolant $I = Itp(\varphi, \psi)$ is a space explanation of c and it satisfies $\varphi \implies I$ (and hence $\mathbb{F}_\varphi \subseteq \mathbb{F}_I$).

Hence, we have a universal means of generalization of existing space explanations, still guaranteeing the classification. Although space explanations are not tied to specific data samples, formulas $\varphi_{\mathbf{s}}$ can be used as a starting space explanation, for which the interpolants are computed quickly because all input variables are fixed to concrete values. This offers the following improvements over existing methods:

1. The captured feature relationships stem from a mathematical *proof* of the classification, hence providing *meaningful* information.

¹Given an unsatisfiable formula $A \wedge B$, a *Craig interpolant* [12] is a formula I such that A implies I , $I \wedge B$ is unsatisfiable, and I uses only the common variables of A and B . We denote interpolant I computed by an interpolation procedure Itp from formulas A and B by $I = Itp(A, B)$.

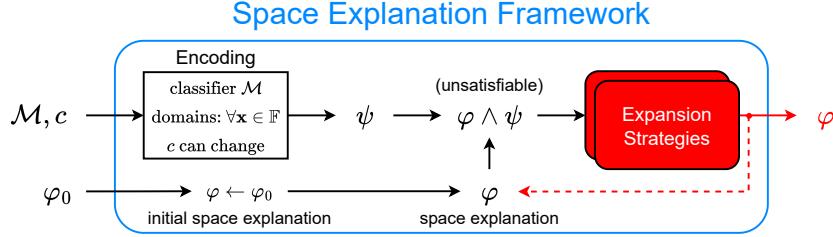


Figure 2: Algorithmic framework: starting from an initial space explanation φ_0 , the impact space is expanded and a logically weaker explanation φ is produced. The user may select from multiple strategies for further optimization (cf. the dashed arrow).

2. The concept is *flexible*, offering to use *arbitrary* Craig interpolation algorithms. The resulting explanations exhibit various logical strengths and forms.
3. It is possible to further *generalize* any *existing* space explanation.

Interpolation algorithms can benefit from special cases [19] where a formula can be partitioned. We address just one example when a space explanation can be split into two parts $\varphi_A \wedge \varphi_B$: given a Craig interpolation procedure Itp with $I = Itp(\varphi_A, \varphi_B \wedge \psi)$, then $I \wedge \varphi_B$ is a logically weaker space explanation. We used this technique to identify pair feature relationships in Figure 1.

Figure 2 gives a high-level representation of a fully functioning and flexible algorithmic *framework* that provides expansion of space explanations using a collection of *strategies* listed below.

- **Generalize (G).** Compute Craig interpolants on an arbitrary space explanation. We use the algorithms based on Farkas' lemma (Itp_F) or their logically stronger *decomposing* variants [7] (Itp_{DF}), the dual versions of these algorithms (Itp'_F , Itp'_{DF}) to yield weaker interpolants [15], and an algorithm [3] (Itp_f) parametrized by a rational factor $f \in [0, 1]$ with the logical strength in between Itp_F and Itp'_F . These arithmetic interpolation algorithms are combined with McMillan's propositional interpolation algorithm [33].
- **Reduce (R).** Weaken the formula and reduce size by computing an unsatisfiable core. To get irreducible explanations (R_{min}), we use exhaustive minimization which may introduce significant overhead.
- **Capture (C).** Partition an interval-like formula (e.g. a sample) and **Generalize**. Weaken only part φ_A with constraints over at least one of the given features, hence capturing their mutual relationships.

Strategy **Generalize** and **Capture** is often followed by **Reduce** to simplify the formulas from unnecessary constraints.

4 Experimental Evaluation

We showcase² the computation of space explanations for selected trained neural networks and demonstrate that they are substantially more expressive and meaningful since our approach captures feature relationships and approximates decision boundaries. We implemented the novel Space Explanation Framework in a prototype tool $S_{\text{P}}\text{EX}_{\text{PL}}\text{AI}_N$ ³, focusing on QF_LRA logic, on top of the OPEN-SMT2 solver [8, 22] which is an interpolating solver that comes with a set of techniques assembled into the integral framework, combining SMT solving with the computation of the Craig interpolants of various size and strength [2, 3, 7, 9, 21, 40]. We evaluated the following NN datasets and models:

- Heart attack dataset [25] focuses on predicting the risk of heart attacks based on various medical indicators of patients, it contains 13 input features and 2 possible classification outcomes: high or low risk. We trained the NN using one hidden layer with 50 neurons, and used a dataset with 303 sample points.
- Obesity dataset [35] provides data for estimating obesity levels in individuals based on their eating habits and physical condition, resulting in 15 input features and 7 classes. We trained the model using 3 hidden layers with 10, 20, and 10 neurons, respectively, and used a dataset with 50 sample points.
- MNIST dataset [13] is a collection of grayscale images of handwritten digits (0–9) with 784 inputs, commonly used within image classification tasks as a reference for training, evaluation, and verification of machine learning models. We trained the model using 784 inputs and a hidden layer with 200 neurons, and used a dataset with 50 sample points.

Instances of interpolation algorithms use the notation $Itp_{DF} \mapsto \mathbf{stronger}$, $Itp_F \mapsto \mathbf{strong}$, $Itp'_F \mapsto \mathbf{weak}$, $Itp'_{DF} \mapsto \mathbf{weaker}$, and $Itp_f \mapsto \mathbf{mid}$ with $f := 0.5$. The evaluation uses 2 hour *time-out* to compute all explanations for a dataset and an explanation setup, and was performed on a Linux 5.4 machine with 256 GB physical memory and AMD® EPYC 7452 32-core CPU.

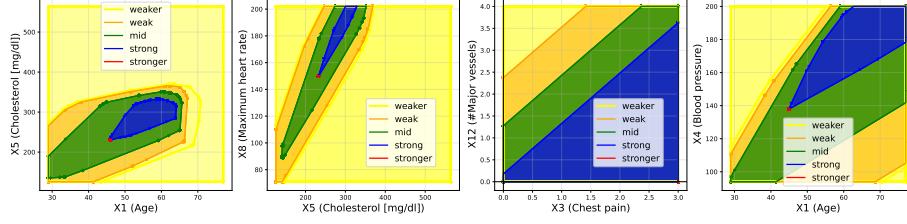
Evaluation of Strategies

Instantiations of strategy **Generalize (G)** provide flexibility of covering spaces with various size, illustrated in Figure 3a on selected sample points from the dataset. We resolve the issue of visualizing high-dimensional spaces by selecting candidate pairs of features from the heart attack dataset and plotting the projection⁴ of the impact spaces into two dimensions. The examples reveal

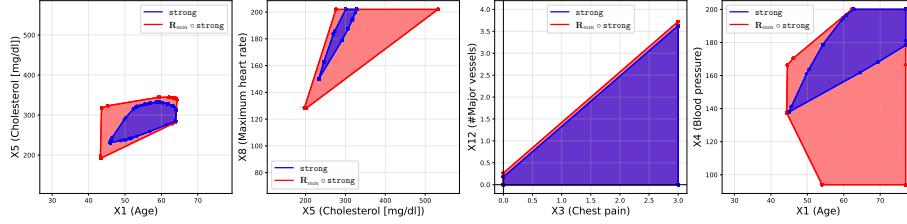
²All table contents are reproducible: <https://doi.org/10.5281/zenodo.1549012>

³<https://github.com/usi-verification-and-security/spexplain>

⁴Projections offer a visual intuitive understanding of the geometrical differences between impact spaces. While not capturing the complete information contained in higher-dimensional spaces, they are still more informative than simple slices.



(a) Generalize (\mathbf{G}): Interpolation-based explanation impact spaces (projections)



(b) Reduce ($\mathbf{R}_{\min} \circ \mathbf{G}$): \mathbf{G} -strong explanations and their exhaustive simplifications (projections)

Figure 3: Projections into selected pairs of features of interpolation-based explanations (\mathbf{G}) of the heart-attack risk model, possibly combined with reduction (\mathbf{R}_{\min}).

Table 1: Average performance of strategy \mathbf{G} using different Itp algorithms.

Itp algorithm	Heart attack			Obesity			MNIST		
	relaxed	#terms	time[s]	relaxed	#terms	time[s]	relaxed	#terms	time[s]
stronger	0%	20.1	0.03	0%	29.0	0.30	0%	927.3	9.17
strong	97%	51.0	0.04	72%	45.8	0.30	100%	209.0	10.12
mid	100%	51.0	0.04	100%	45.8	0.34	100%	209.0	10.25
weak	100%	51.0	0.04	100%	46.1	0.30	100%	209.0	10.42
weaker	100%	198.2	0.04	100%	64.1	0.42	100%	67330.6	10.55

the similarity of feature relations despite aiming at different logical strengths. However, we spot that **weaker** here always covers the whole projection⁵, while **stronger** in turn results just in the original sample point. This is no exception: Table 1 reports that **stronger** relaxes no features at all (i.e. each feature is still fixed to the original value), producing way too strong interpolants. Algorithm **strong** sometimes does not relax all features. Next, the table reports the average size of the resulting formulas term-wise (the number of in/equalities). The average runtimes per sample point are rather small and do not vary among the approaches, but are sensitive to more complex models.

Strategy **Reduce** (\mathbf{R} , \mathbf{R}_{\min}) introduces a trade-off between formula simplification and runtime overhead when applied on top of **Generalize** (e.g., $\mathbf{R} \circ \mathbf{G}$), using an unsatisfiable core (\mathbf{R}) or an irreducible one (\mathbf{R}_{\min}). Figure 3a demon-

⁵Notably, the explanation does not cover the whole feature space. However, high-dimensional projections (e.g., from 13D to 2D) collapse information from other dimensions, making explanations appear larger.

Table 2: Time overhead and impact of **Reduce** on top of **Generalize** ($\mathbf{R} \circ \mathbf{G}$).

Itp algorithm	#terms			time [s]		
	-	\mathbf{R}	\mathbf{R}_{min}	-	\mathbf{R}	\mathbf{R}_{min}
Heart attack	stronger	20.1	20.1	9.5	0.03	0.03
	strong	51.0	44.6	3.8	0.04	0.83
	mid	51.0	39.4	4.6	0.04	0.66
	weak	51.0	38.8	6.4	0.04	0.56
	weaker	198.2	197.9	25.9	0.04	0.08
Obesity	stronger	29.0	29.0	X	0.30	0.38
	strong	45.8	41.7	X	0.30	7.94
	mid	45.8	42.7	X	0.34	20.62
	weak	46.1	41.7	X	0.30	35.37
	weaker	64.1	58.0	X	0.42	23.44
MNIST	stronger	927.3	927.3	X	9.17	10.45
	strong	209.0	X	X	10.12	X
	mid	209.0	X	X	10.25	X
	weak	209.0	X	X	10.42	X
	weaker	67330.6	X	X	10.55	X

strates side by side with Figure 3b that the projection of **strong** is further expanded by \mathbf{R}_{min} (\mathbf{R} expands none of the four presented **strong** projections), but in different directions compared to weaker algorithms. We confirmed this phenomenon by running extensive subset-comparisons which in cases such as $(\text{strong}, \mathbf{R}_{min})$ vs. $(\text{weak}, -)$ often yielded not comparable (NC) results: the spaces intersect but none of them subsumes the other. Table 2 shows that \mathbf{R}_{min} significantly reduces the formula but with a high cost, while \mathbf{R} offers a convenient balance. The reduction times-out (X) with more complex models.

Figure 4a follows with a comparison of projections between strategy **G** (cf. Figure 3a), specifically selecting **weak** algorithm that produces large yet convex impact spaces, and an approach that is based on intervals, denoted as **I**, that computes the explanations on top of an irreducible abductive explanation, denoted by **A**, resulting in $\mathbf{I} \circ \mathbf{A}$. Since the state-of-the-art-approach [24] is implemented for decision trees and not for NNs, we implemented the specialized computation of interval explanations using the MARABOU verifier and a strict limit on the number of attempts of particular relaxations. We concentrate on comparison using the heart attack model. The projections of interval-based explanations form rectangles, lines, or even single points due to the lack of their expressivity. In contrast, **G** explanations cover larger areas with less limited shapes, yet do not entirely subsume interval explanations: Even if a projection is subsumed, it does not mean that the space is subsumed in the other dimensions as well. While the intervals are being relaxed in a certain order, hence strictly preferring certain directions over others, the interpolation aims at more general expansions. Furthermore, the expansion of the interval explanations is limited by decision boundaries in the other dimensions. This observation is

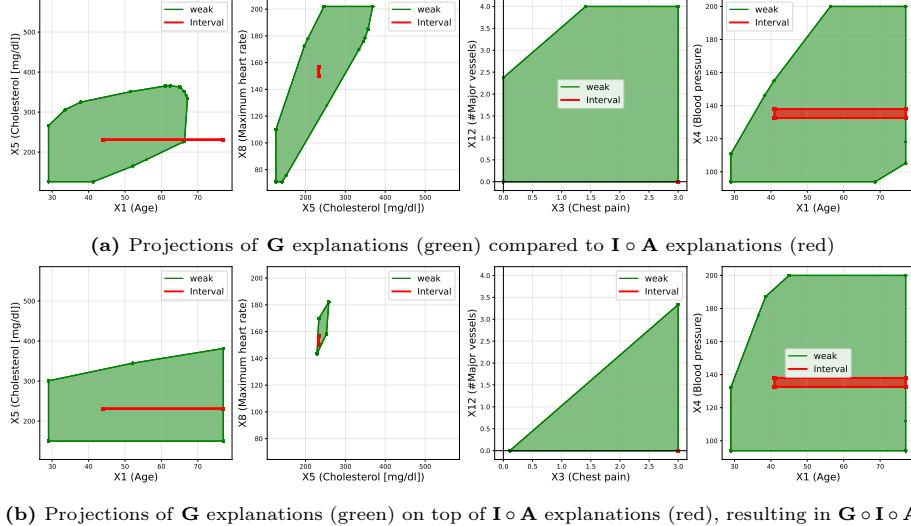


Figure 4: Comparing projections of interpolation-based (**Generalize**, **G**) and interval explanations (**I** \circ **A**) into selected pairs of features of the heart-attack risk model.

Table 3: Average performance of **G** vs. **A** and **I** strategies in heart-attack model.

	relaxed	#terms	#solver calls	time [s]
A	38%	8.1	13	0.08
I \circ A	79%	9.3	40.4	0.53
G	100%	51.0	1	0.04
G \circ A	100%	45.3	1	0.39
G \circ I \circ A	100%	63.5	1	2.53

confirmed by pairwise subset-comparisons, consistently arriving at **NC** results.

Table 3 gives comparison of the average performance of the approaches (similarly as in Table 1), including irreducible abductive explanations (**A**). **G** explanations (row #3) are computed even faster than **A** explanations, requiring just one call to the logical solver. Interval explanations did not always succeed in relaxing all features. Nevertheless, the formulas are simpler when based on intervals or samples. Formulas from **G** could be reduced but would exceed the computation time of intervals. The observations remain similar when using other algorithms than **weak**. Figure 4b further illustrates the potentials of strategy **G** when running on top of arbitrary existing explanations, such as interval explanations (i.e., **G** \circ **I** \circ **A**), yielding strictly larger spaces (confirmed by subset-comparison checks). Nonetheless, the projections of the interpolants in Figures 4a vs. 4b (i.e., **G** vs. **G** \circ **I** \circ **A**) sometimes differ due to more guided constraints induced by the intervals. Next, we revisit Table 3 on rows #4–5, including **G** \circ **A** as well. The observations remain similar except that when gen-

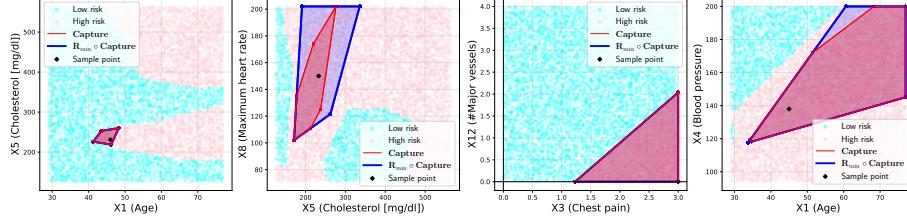


Figure 5: Approximation of decision boundaries and feature relations using strategies **C** and $\mathbf{R}_{min} \circ \mathbf{C}$ within selected pairs of features of the heart-attack risk model.

Table 4: Quantitative subset-comparison of **C** vs. **G** explanations, combined with **R** or \mathbf{R}_{min} , ranging over selected features where the others are fixed to the original values.

Features	C	G (sliced)	\supset	$=$	\subset	NC
x_1, x_5	$\mathbf{R} \circ \mathbf{C}$ vs.	$\mathbf{R} \circ \mathbf{G}$	8%	43%	41%	7%
	$\mathbf{R}_{min} \circ \mathbf{C}$ vs.	$\mathbf{R}_{min} \circ \mathbf{G}$	65%	0%	0%	35%
x_3, x_{12}	$\mathbf{R} \circ \mathbf{C}$ vs.	$\mathbf{R} \circ \mathbf{G}$	2%	92%	5%	1%
	$\mathbf{R}_{min} \circ \mathbf{C}$ vs.	$\mathbf{R}_{min} \circ \mathbf{G}$	89%	5%	0%	6%
x_1, x_4	$\mathbf{R} \circ \mathbf{C}$ vs.	$\mathbf{R} \circ \mathbf{G}$	9%	54%	31%	6%
	$\mathbf{R}_{min} \circ \mathbf{C}$ vs.	$\mathbf{R}_{min} \circ \mathbf{G}$	84%	0%	0%	16%

eralizing on top of a more general starting point, the runtime increases. Still, our generalization of **A** explanations runs faster than if using intervals (i.e., $\mathbf{I} \circ \mathbf{A}$ vs. $\mathbf{G} \circ \mathbf{A}$).

Strategy **Capture (C)** directly aims at identifying relationships among selected features and consequently at approximating decision boundaries. The generalization is guided because it focuses on selected dimensions and picks a slice of the feature space, leaving all other features fixed to the original values. Although such limited exploration does not reveal anything about other dimensions, it extracts partial information helpful to understand the classification. Moreover, it is easy to sample two or three-dimensional slices and compare the explanations with estimated class spaces and decision boundaries. Figure 5 shows the explanations of the sample points and feature pairs as in Figures 3 and 4, this time using **C** instead of **G**, sticking to algorithm **weak**. The interpolation captures even non-trivial decision boundaries and some of the relations intuitively resemble real-world phenomena. For example, the plot on the right identifies a high risk of heart attack according to increasing age and decreasing blood pressure.

Using **Reduce (R, R_{min})** is especially useful for explanations produced by strategy **C** to enhance their interpretability. Furthermore, the reduction is often more efficient than when applied to **G**. Table 4 shows the quantitative subset-comparison, that is, how many explanations in percentage exhibited the relation superset (\supset), equivalent ($=$), subset (\subset), or not comparable (**NC**), between **C** and **G** explanations that have been reduced and while focusing just on the slice of selected pairs of features (cf. Figures 3–5). With no reduction, the slices

Table 5: Average performance of strategies **C** and **G** from Table 4 possibly in combination with **Reduce** (**R** or **R_{min}**).

Reduce	Heart attack			Obesity			MNIST			
	C	G (sliced)	#terms time[s]	C	G (sliced)	#terms time[s]	C	G (sliced)	#terms time[s]	
—	61.2	0.04	62.0	0.04	59.1	0.33	59.1	0.31	938.9	9.65
R	47.0	0.09	49.8	0.56	41.9	1.15	54.7	35.3	834.7	47.1
R_{min}	9.4	2.52	17.4	7.94	X	X	X	X	X	X

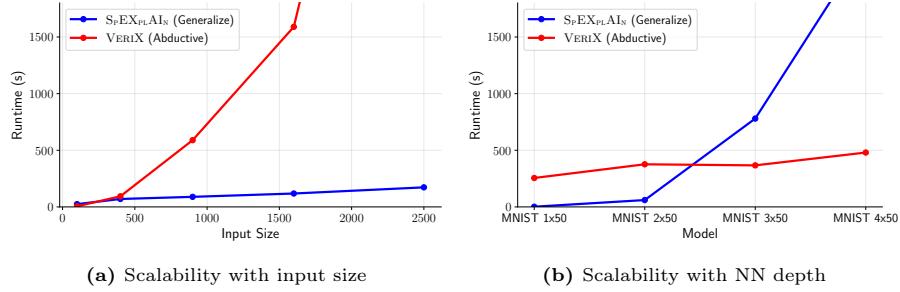


Figure 6: Scalability comparison of $S_pEXPLAI_N$ and $VERIX$ on MNIST

are equivalent. Yet, the formulas differ, because strategy **C** separates the fixed features from the focused ones⁶. Consequently, the opportunities for reductions offered by the structure of the formulas are different. While strategy **R** usually expands the explanations better when applied to **G** (i.e. $\mathbf{R} \circ \mathbf{G}$), $\mathbf{R}_{min} \circ \mathbf{C}$ always produced a larger space within the slice than $\mathbf{R}_{min} \circ \mathbf{G}$, if excluding NC cases. The results for x_5, x_8 are almost the same as for x_1, x_5 . Finally, Table 5 shows that the simplification is more efficient in terms of runtime and #terms when using the focused strategy **C** than if interpolating the whole formula with **G**.

Scalability

We conducted two experiments to evaluate how our prototype tool $S_pEXPLAI_N$ scales with increasing input size and depth of the NN using MNIST. We compare our method with $VERIX$ [44], a state-of-the-art abductive explanation algorithm⁷. While the approaches generate different types of explanations and are not directly comparable, their relative scalability can still be compared.

The first experiment shows how the runtime grows with input size. For that, we used a fixed NN architecture with two hidden layers of size 50⁸, and resized MNIST images to input dimensions 10×10 , 20×20 , 30×30 , 40×40 , and 50×50 . Figure 6a shows how the runtime grows with the input size for computing **G** explanations and abductive explanations: the runtime of $VERIX$

⁶Example: **G** sliced to x_1, x_2 yields $(2x_1 + x_3 \geq 7 \wedge x_1 - x_2 + x_3 \geq 2) \wedge x_3 = 1$ which is equivalent to $(x_1 \geq 3 \wedge x_1 - x_2 \geq 1) \wedge x_3 = 1$ produced by **C** of x_1, x_2 .

⁷<https://github.com/NeuralNetworkVerification/VeriX>, commit b6b2cc0

⁸A smaller network was used to reduce the number of time-outs.

increases steeply, while **Generalize** remains relatively stable. The difference is caused by the fact that VERIX performs one verification query per feature, while our method requires only a single query to generate an explanation.

The next experiment examines scalability with respect to model depth. We trained NNs with 1 to 4 hidden layers, each having 50 neurons. Figure 6b shows that the runtime of VERIX grows moderately with the number of layers, while the runtime of **Generalize** increases more significantly with deeper networks. This is likely because, unlike VERIX, our prototype tool does not apply any NN-specific optimizations within its underlying constraint solver. Addressing this limitation is left for future work.

Lessons Learned

The following table summarizes our observations regarding the computation, interpretation, simplification, and comparison of explanations.

Observations	Solution
Problem: Explanations produced by Generalize are too complicated to read.	
<ul style="list-style-type: none"> 1. <code>simplify</code> command of Z3 or cvc5 SMT solvers is insufficient. 2. Reduce simplifies conjunctive formulas, often surprisingly well. 3. McMillan interpolation algorithm extends formulas into conjunctions and its dual into disjunctions. 	Use the non-dual algorithm and simplify the explanations using Reduce if needed.
Problem: Information regarding just a few features is difficult to extract, even if we use Reduce .	
<ul style="list-style-type: none"> 1. Partitioning interval-like formulas separates selected features. 	Use Capture and Reduce .
Problem: It is not clear which interpolation algorithm to choose.	
<ul style="list-style-type: none"> 1. The yielded formulas differ in logical strength and length. 2. The weaker the formula, the larger the space covered, offering the best opportunity to e.g. approximate decision boundaries. 3. Smaller or more focused formulas might be easier to interpret. 	Use multiple algorithms depending on the current application and flexibly pick suitable outcomes.
Problem: It is not clear how to visualize explanations with high-dimensional impact spaces.	
<ul style="list-style-type: none"> 1. Projections (cf. Figure 3) may yield too large area since the information from the other dimensions is collapsed. 2. Slices fix all other dimensions (cf. Figure 5), but conclusions on those dimensions are very limited. However, they allow direct comparison with decision boundaries or their sampling. 	Use slices to show local information and to compare with decision boundaries. Use projections to compare the robustness of explanations.
Problem: Sometimes, quantitative and more rigorous comparisons than visualization are needed.	
<ul style="list-style-type: none"> 1. Computing the volume of high-dimensional spaces is non-trivial. 2. It is easy to check if a space explanation is subsumed by another using the implication of the formulas (in conjunction with ψ_D). 3. Spaces often intersect but are not entirely subsumed by another. 	Estimate the quality using multiple metrics. If needed, include a visual comparison.

Idea: What if we compared only selected features?	
<ol style="list-style-type: none"> 1. Computing the volume of 2D or 3D spaces is feasible. 2. Quantifier elimination yields exact projections but is expensive. 3. Projections can be approximated via linear programming. 4. Whole explanations can be compared by exhaustively comparing their projections into all particular pairs (or triplets) of features. 	Comparing only selected features is a simpler problem and may also enable more thorough comparisons.

5 Conclusion and Future Work

This paper presented a novel concept of provably correct, logic-based space explanations of the classification process of neural networks. The explanations are associated with complex-shaped spaces and capture relations among the features that stem from mathematical proofs, substantially improving the approximation of decision boundaries over existing methods. The Space Explanations concept is supported by a flexible framework of algorithms including efficient Craig interpolation-based techniques and unsatisfiable core extraction to compute an extensive range of different yet meaningful explanations. On real-world neural networks trained on practical datasets, we performed a series of experiments and computed explanations of different quality. The evaluation of our case studies confirms that our algorithms yield explanations that are more general than existing explanations. We shared lessons learned during the extensive experimentation with the new kind of explanations.

In future work, we will develop an algorithm to approximate decision boundaries and to identify reasons for misclassifications across clusters of the feature space. We will improve the scalability of our tool $S_{\text{P}}\text{EX}_{\text{PL}}\text{AI}_{\text{N}}$ by using optimizations tailored to NNs, and aim to handle other NN structures, such as convolutional NNs. Finally, we will apply our method to analyze how decisions evolve across the hidden layers of the network.

Acknowledgements. This work was conducted as part of the “Formal Reasoning on Neural Networks” project funded by the Hasler Foundation, Switzerland.

References

- [1] Albargouthi, A., Dillig, I., Gurfinkel, A.: Maximal Specification Synthesis. In: Bodík, R., Majumdar, R. (eds.) Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016. pp. 789–801. ACM (2016). <https://doi.org/10.1145/2837614.2837628>

- [2] Alt, L., Fedukovich, G., Hyvärinen, A.E.J., Sharygina, N.: A proof-sensitive approach for small propositional interpolants. In: Gurfinkel, A., Seshia, S.A. (eds.) Verified Software: Theories, Tools, and Experiments - 7th International Conference, VSTTE 2015, San Francisco, CA, USA, July 18-19, 2015. Revised Selected Papers. Lecture Notes in Computer Science, vol. 9593, pp. 1–18. Springer (2015). https://doi.org/10.1007/978-3-319-29613-5_1
- [3] Alt, L., Hyvärinen, A.E.J., Sharygina, N.: LRA interpolants from no man's land. In: Strichman, O., Tzoref-Brill, R. (eds.) Hardware and Software: Verification and Testing - 13th International Haifa Verification Conference, HVC 2017, Haifa, Israel, November 13-15, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10629, pp. 195–210. Springer (2017). https://doi.org/10.1007/978-3-319-70389-3_13
- [4] Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. PLOS One **10**(7) (2015)
- [5] Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., Müller, K.: How to Explain Individual Classification Decisions. J. Mach. Learn. Res. **11**, 1803–1831 (2010). <https://doi.org/10.5555/1756006.1859912>, <https://dl.acm.org/doi/10.5555/1756006.1859912>
- [6] Barbosa, H., Barrett, C.W., Brain, M., Kremer, G., Lachnitt, H., Mann, M., Mohamed, A., Mohamed, M., Niemetz, A., Nötzli, A., Ozdemir, A., Preiner, M., Reynolds, A., Sheng, Y., Tinelli, C., Zohar, Y.: cvc5: A Versatile and Industrial-Strength SMT Solver. In: Fisman, D., Rosu, G. (eds.) Tools and Algorithms for the Construction and Analysis of Systems - 28th International Conference, TACAS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13243, pp. 415–442. Springer (2022). https://doi.org/10.1007/978-3-030-99524-9_24
- [7] Blichá, M., Hyvärinen, A.E.J., Kofron, J., Sharygina, N.: Decomposing Farkas interpolants. In: Vojnar, T., Zhang, L. (eds.) Tools and Algorithms for the Construction and Analysis of Systems - 25th International Conference, TACAS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11427, pp. 3–20. Springer (2019). https://doi.org/10.1007/978-3-030-17462-0_1

- [8] Bruttomesso, R., Pek, E., Sharygina, N., Tsitovich, A.: The OpenSMT Solver. In: Esparza, J., Majumdar, R. (eds.) Tools and Algorithms for the Construction and Analysis of Systems, 16th International Conference, TACAS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6015, pp. 150–153. Springer (2010). https://doi.org/10.1007/978-3-642-12002-2_12
- [9] Bruttomesso, R., Rollini, S., Sharygina, N., Tsitovich, A.: Flexible interpolation with local proof transformations. In: Scheffer, L., Phillips, J.R., Hu, A.J. (eds.) 2010 International Conference on Computer-Aided Design, ICCAD 2010, San Jose, CA, USA, November 7-11, 2010. pp. 770–777. IEEE (2010). <https://doi.org/10.1109/ICCAD.2010.5654297>
- [10] Christ, J., Hoenicke, J., Nutz, A.: SMTInterpol: An Interpolating SMT Solver. In: Donaldson, A.F., Parker, D. (eds.) Model Checking Software - 19th International Workshop, SPIN 2012, Oxford, UK, July 23-24, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7385, pp. 248–254. Springer (2012). https://doi.org/10.1007/978-3-642-31759-0_19
- [11] Cimatti, A., Griggio, A., Schaafsma, B.J., Sebastiani, R.: The MathSAT5 SMT Solver. In: Piterman, N., Smolka, S.A. (eds.) Tools and Algorithms for the Construction and Analysis of Systems - 19th International Conference, TACAS 2013, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013, Rome, Italy, March 16-24, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7795, pp. 93–107. Springer (2013). https://doi.org/10.1007/978-3-642-36742-7_7
- [12] Craig, W.: Three Uses of the Herbrand-Gentzen Theorem in Relating Model Theory and Proof Theory. In: J. of Symbolic Logic. vol. 22, pp. 269–285 (1957). <https://doi.org/10.2307/2963594>
- [13] Deng, L.: The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. IEEE Signal Processing Magazine **29**(6), 141–142 (2012). <https://doi.org/10.1109/MSP.2012.2211477>
- [14] Dillig, I., Dillig, T., Li, B., McMillan, K.L.: Inductive invariant generation via abductive inference. In: Hosking, A.L., Eugster, P.T., Lopes, C.V. (eds.) Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages & Applications, OOPSLA 2013, part of SPLASH 2013, Indianapolis, IN, USA, October 26-31, 2013. pp. 443–456. ACM (2013). <https://doi.org/10.1145/2509136.2509511>

- [15] D'Silva, V.V., Kroening, D., Purandare, M., Weissnabcher, G.: Interpolant Strength. In: Barthe, G., Hermenegildo, M.V. (eds.) Verification, Model Checking, and Abstract Interpretation, 11th International Conference, VMCAI 2010, Madrid, Spain, January 17-19, 2010. Proceedings. Lecture Notes in Computer Science, vol. 5944, pp. 129–145. Springer (2010). https://doi.org/10.1007/978-3-642-11319-2_12
- [16] Dutertre, B.: Yices 2.2. In: Biere, A., Bloem, R. (eds.) Computer Aided Verification - 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18-22, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8559, pp. 737–744. Springer (2014). https://doi.org/10.1007/978-3-319-08867-9_49
- [17] Erhan, D., Bengio, Y., Courville, A., Vincent, P.: Visualizing Higher-layer Features of a Deep Network. Tech. rep., University of Montreal (2009)
- [18] Group, E.E.: Ethics Guidelines for Trustworthy AI. Available online: https://ec.europa.eu/newsroom/dae/document.cfm?doc_id=60419
- [19] Gurfinkel, A., Rollini, S.F., Sharygina, N.: Interpolation Properties and SAT-Based Model Checking. In: Hung, D.V., Ogawa, M. (eds.) Automated Technology for Verification and Analysis - 11th International Symposium, ATVA 2013, Hanoi, Vietnam, October 15-18, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8172, pp. 255–271. Springer (2013). https://doi.org/10.1007/978-3-319-02444-8_19
- [20] Heaton, J.: Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning - The MIT Press, 2016, 800 pp, ISBN: 0262035618. Genet. Program. Evolvable Mach. **19**(1-2), 305–307 (2018). <https://doi.org/10.1007/S10710-017-9314-Z>, <https://doi.org/10.1007/s10710-017-9314-z>
- [21] Hyvärinen, A.E.J., Alt, L., Sharygina, N.: Flexible interpolation for efficient model checking. In: Kofron, J., Vojnar, T. (eds.) Mathematical and Engineering Methods in Computer Science - 10th International Doctoral Workshop, MEMICS 2015, Telč, Czech Republic, October 23-25, 2015, Revised Selected Papers. Lecture Notes in Computer Science, vol. 9548, pp. 11–22. Springer (2015). https://doi.org/10.1007/978-3-319-29817-7_2, https://doi.org/10.1007/978-3-319-29817-7_2
- [22] Hyvärinen, A.E.J., Marescotti, M., Alt, L., Sharygina, N.: OpenSMT2: An SMT Solver for Multi-core and Cloud Computing. In: Creignou, N., Berre, D.L. (eds.) Theory and Applications of Satisfiability Testing - SAT 2016 - 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings. Lecture Notes in Computer Science, vol. 9710, pp. 547–553.

Springer (2016). https://doi.org/10.1007/978-3-319-40970-2_35

- [23] Ignatiev, A., Narodytska, N., Marques-Silva, J.: Abduction-Based Explanations for Machine Learning Models. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. pp. 1511–1519. AAAI Press (2019). <https://doi.org/10.1609/AAAI.V33I01.33011511>, <https://doi.org/10.1609/aaai.v33i01.33011511>
- [24] Izza, Y., Ignatiev, A., Stuckey, P.J., Marques-Silva, J.: Delivering Inflated Explanations. In: Wooldridge, M.J., Dy, J.G., Natarajan, S. (eds.) Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada. pp. 12744–12753. AAAI Press (2024). <https://doi.org/10.1609/AAAI.V38I11.29170>, <https://doi.org/10.1609/aaai.v38i11.29170>
- [25] Janosi, A., Steinbrunn, W., Pfisterer, M., Detrano, R.: Heart Disease. UCI Machine Learning Repository (1989). <https://doi.org/10.24432/C5P338>, DOI: <https://doi.org/10.24432/C52P4X>
- [26] Katz, G., Huang, D.A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljic, A., Dill, D.L., Kochenderfer, M.J., Barrett, C.W.: The Marabou Framework for Verification and Analysis of Deep Neural Networks. In: Dillig, I., Tasiran, S. (eds.) Computer Aided Verification - 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11561, pp. 443–452. Springer (2019). https://doi.org/10.1007/978-3-030-25540-4_26, https://doi.org/10.1007/978-3-030-25540-4_26
- [27] König, M., Bosman, A.W., Hoos, H.H., van Rijn, J.N.: Critically Assessing the State of the Art in Neural Network Verification. J. Mach. Learn. Res. **25**, 12:1–12:53 (2024), <https://jmlr.org/papers/v25/23-0119.html>
- [28] Labbaf, F., Kolárik, T., Blichá, M., Fedyukovich, G., Wand, M., Sharygina, N.: Space explanations of neural network classification. In: Piskac, R., Rakamarić, Z. (eds.) CAV. pp. 287–303. Springer Nature Switzerland, Cham (2025)
- [29] Lundberg, S.M., Lee, S.: A Unified Approach to Interpreting Model Predictions. In: Guyon, I., von Luxburg, U., Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA.

- pp. 4765–4774 (2017), <https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html>
- [30] Mahendran, A., Vedaldi, A.: Understanding deep image representations by inverting them. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015. pp. 5188–5196. IEEE Computer Society (2015). <https://doi.org/10.1109/CVPR.2015.7299155>, <https://doi.org/10.1109/CVPR.2015.7299155>
 - [31] Malfa, E.L., Michelmore, R., Zbrzezny, A.M., Paoletti, N., Kwiatkowska, M.: On Guaranteed Optimal Robust Explanations for NLP Models. In: Zhou, Z. (ed.) Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021. pp. 2658–2665. ijcai.org (2021). <https://doi.org/10.24963/IJCAI.2021/366>, <https://doi.org/10.24963/ijcai.2021/366>
 - [32] Marques-Silva, J.: Logic-Based Explainability in Machine Learning. In: Bertossi, L.E., Xiao, G. (eds.) Reasoning Web. Causality, Explanations and Declarative Knowledge - 18th International Summer School 2022, Berlin, Germany, September 27-30, 2022, Tutorial Lectures. Lecture Notes in Computer Science, vol. 13759, pp. 24–104. Springer (2022). https://doi.org/10.1007/978-3-031-31414-8_2, https://doi.org/10.1007/978-3-031-31414-8_2
 - [33] McMillan, K.L.: Interpolation and SAT-Based Model Checking. In: Jr., W.A.H., Somenzi, F. (eds.) Computer Aided Verification, 15th International Conference, CAV 2003, Boulder, CO, USA, July 8-12, 2003, Proceedings. Lecture Notes in Computer Science, vol. 2725, pp. 1–13. Springer (2003). https://doi.org/10.1007/978-3-540-45069-6_1, https://doi.org/10.1007/978-3-540-45069-6_1
 - [34] de Moura, L.M., Bjørner, N.S.: Z3: an efficient SMT solver. In: Ramakrishnan, C.R., Rehof, J. (eds.) Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings. Lecture Notes in Computer Science, vol. 4963, pp. 337–340. Springer (2008). https://doi.org/10.1007/978-3-540-78800-3_24, https://doi.org/10.1007/978-3-540-78800-3_24
 - [35] Palechor, F.M., Manotas, A.D.I.H.: Estimation of Obesity Levels Based On Eating Habits and Physical Condition. UCI Machine Learning Repository (2019). <https://doi.org/10.24432/C5H31Z>, <https://doi.org/10.24432/C5H31Z>, DOI: <https://doi.org/10.24432/C5H31Z>
 - [36] Prabhu, S., D’Souza, D., Chakraborty, S., Venkatesh, R., Fedyukovich, G.: Weakest Precondition Inference for Non-Deterministic Linear Array

- Programs. In: Finkbeiner, B., Kovács, L. (eds.) Tools and Algorithms for the Construction and Analysis of Systems - 30th International Conference, TACAS 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part II. Lecture Notes in Computer Science, vol. 14571, pp. 175–195. Springer (2024). https://doi.org/10.1007/978-3-031-57249-4_9
- [37] Prabhu, S., Fedyukovich, G., D’Souza, D.: Maximal Quantified Precondition Synthesis for Linear Array Loops. In: Weirich, S. (ed.) Programming Languages and Systems - 33rd European Symposium on Programming, ESOP 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part II. Lecture Notes in Computer Science, vol. 14577, pp. 245–274. Springer (2024). https://doi.org/10.1007/978-3-031-57267-8_10
 - [38] Ribeiro, M.T., Singh, S., Guestrin, C.: "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In: Krishnapuram, B., Shah, M., Smola, A.J., Aggarwal, C.C., Shen, D., Rastogi, R. (eds.) Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016. pp. 1135–1144. ACM (2016). <https://doi.org/10.1145/2939672.2939778>
 - [39] Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: High-Precision Model-Agnostic Explanations. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018. pp. 1527–1535. AAAI Press (2018). [https://doi.org/10.1609/aaai.v32i1.11491](https://doi.org/10.1609/AAAI.V32I1.11491)
 - [40] Rollini, S.F., Alt, L., Fedyukovich, G., Hyvärinen, A.E.J., Sharygina, N.: Periplo: A framework for producing effective interpolants in sat-based software verification. In: McMillan, K.L., Middeldorp, A., Voronkov, A. (eds.) Logic for Programming, Artificial Intelligence, and Reasoning - 19th International Conference, LPAR-19, Stellenbosch, South Africa, December 14-19, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8312, pp. 683–693. Springer (2013). https://doi.org/10.1007/978-3-642-45221-5_45
 - [41] Seshia, S.A., Sadigh, D.: Towards Verified Artificial Intelligence. CoRR (2016), <http://arxiv.org/abs/1606.08514>

- [42] Shih, A., Choi, A., Darwiche, A.: A Symbolic Approach to Explaining Bayesian Network Classifiers. In: Lang, J. (ed.) Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden. pp. 5103–5111. ijcai.org (2018). <https://doi.org/10.24963/IJCAI.2018/708>
- [43] Simonyan, K., Vedaldi, A., Zisserman, A.: Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In: Bengio, Y., LeCun, Y. (eds.) 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings (2014), <http://arxiv.org/abs/1312.6034>
- [44] Wu, M., Wu, H., Barrett, C.W.: VeriX: Towards Verified Explainability of Deep Neural Networks. In: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023 (2023), http://papers.nips.cc/paper_files/paper/2023/hash/46907c2ff9fafd618095161d76461842-Abstract-Conference.html
- [45] Zeiler, M.D., Fergus, R.: Visualizing and Understanding Convolutional Networks. In: Fleet, D.J., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I. Lecture Notes in Computer Science, vol. 8689, pp. 818–833. Springer (2014). https://doi.org/10.1007/978-3-319-10590-1_53

A Appendix

A.1 Classifying Neural Networks

(Classifying) Neural Networks are a specific class of general classifiers. Following the notation and definitions, we assume that the neural networks perform single-class classification (i.e. each sample point is assigned to exactly one class). A neural network is represented by a graph $G(E, V)$ with the set of edges E and the set of vertices V , representing the neurons and their weighted connections, respectively. The neurons are partitioned into layers V_0, \dots, V_K where each V_k contains neurons $v_i^{(k)}$, $i \in \{1, \dots, n^{(k)}\}$. Layer V_0 is the *input* layer, V_K is the *output* layer, and layers V_1, \dots, V_{K-1} are *hidden* layers. The edges correspond to the pairs connecting every neuron of layer V_k to every neuron in the next layer V_{k+1} , for all $0 \leq k < K$, i.e., edge $e_{i,j}^{(k)}$ connects i -th neuron in layer k with j -th neuron in layer $k+1$. Every edge is assigned a numerical weight $w_{i,j}^{(k)}$. Moreover, every neuron $v_i^{(k)}$ in a hidden layer is assigned a bias $b_i^{(k)}$. Each input

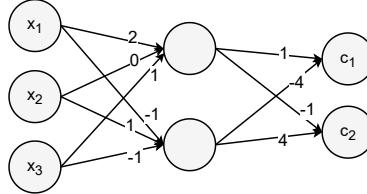


Figure 7: A neural network with one hidden layer

neuron $v_i^{(0)}$ corresponds to feature $i \in \mathcal{F}$, represented by feature variable $x_i \in \mathcal{D}_i$, and each output neuron $v_i^{(K)}$ corresponds to class c_i . Each domain \mathcal{D}_i is either a finite set or a closed interval $\mathcal{D}_i = [L_i, U_i]$ where both the lower and upper bounds are finite⁹.

Given an input sample $\mathbf{s} = (s_1, \dots, s_m)$, the activations $x_i^{(k)}$ for each neuron $v_i^{(k)}$ are computed as follows. In the case of the input layer, $x_i^{(0)} = s_i$. In the case of hidden layers,

$$y_i^{(k)} = \sum_j x_j^{(k-1)} w_{j,i}^{(k-1)} + b_i^{(k)} \quad (1)$$

and $x_i^{(k)} = \text{ReLU}(y_i^{(k)})$, where $\text{ReLU}(x) = \max\{x, 0\}$ is the rectifier function. In the case of the output layer, no activation function is applied¹⁰. The class c_i assigned to input sample \mathbf{s} is given by the output neuron with maximum activation, formally,

$$\kappa(\mathbf{s}) = c_i \iff i = \operatorname{argmax}_j \{y_j^{(K)}\}. \quad (2)$$

Example 1. Figure 7 shows a neural network that maps three-dimensional inputs $\mathbf{x} = (x_1, x_2, x_3)$ to two classes $\{c_1, c_2\}$. The input domains are $\mathcal{D}_i = [0, 4]$, therefore, the feature space corresponds to a cube with edges of length 4. The network consists of 3 input neurons, 2 hidden neurons with ReLU activation, and 2 output neurons. The weights for each edge are labeled accordingly, and for simplicity, biases are set to zero. The output is computed as follows:

$$\begin{aligned} x_1^{(1)} &= \text{ReLU}(2x_1 + x_3) & o_{c_1} &= y_1^{(2)} = x_1^{(1)} - 4x_2^{(1)} \\ x_2^{(1)} &= \text{ReLU}(-x_1 + x_2 - x_3) & o_{c_2} &= y_2^{(2)} = -x_1^{(1)} + 4x_2^{(1)} \end{aligned}$$

where o_{c_1} and o_{c_2} refer to the values of the output neurons. If $o_{c_1} \geq o_{c_2}$, then the classification would be c_1 , otherwise, it would be c_2 , e.g., for input sample $\mathbf{s} = (1, 1, 3)$, $o_{c_1} = 5$ and $o_{c_2} = -5$ and therefore, \mathbf{s} is classified as c_1 .

⁹The restriction to finite intervals is required since a neural network is always trained on a finite set of training data, thus it learns to expect inputs in a finite range; passing an input (far) outside this range inevitably cause unpredictable behavior.

¹⁰During the training process, usually the softmax function is used in the last layer; since this is a monotonic function it can be ignored for the purpose of this paper. For details about the *training*, i.e. the parameter estimation of neural networks, we refer to a large number of available resources, e.g. [20].

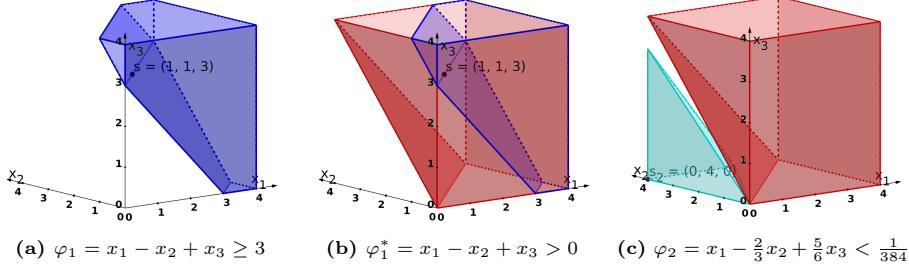


Figure 8: Impact spaces associated with three interpolation-based explanations for the network in Example 1 with the feature space $\mathbb{F} = [0, 4]^3$. Figure (a) represents explanation φ_1 (blue) for the sample point $\mathbf{s} = (1, 1, 3)$, and Figure (b) represents a weaker explanation φ_1^* (red). Figure (c) shows explanation φ_2 (cyan) for a different sample point $\mathbf{s}_2 = (0, 4, 0)$ classified into class c_2 . The area between the impact spaces $\mathbb{F}_{\varphi_1^*}$ and \mathbb{F}_{φ_2} contains the decision boundary.

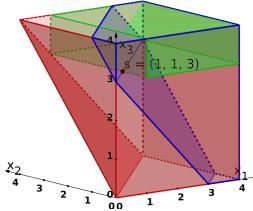


Figure 9: Comparison to interval explanation (green)

A.2 Interpolation-based explanations

This section provides additional details on Section 3.

Example 2. Given the toy neural network in Example 1, we depict three automatically computed interpolation-based explanations in Figure 8. Explanation φ_1 of the sample point $\mathbf{s} = (1, 1, 3)$ classified to class c_1 captures a mutual relationship among all features. The formula resembles the term $x_2^{(1)}$ in Example 1.

Next, we obtained explanation φ_1^* using the same sample point φ_s but a different (logically weaker) interpolation algorithm than with φ_1 . Moreover, φ_1^* can also be computed on top of φ_1 instead of φ_s .

Finally, we obtained explanation φ_2 similarly as φ_1^* but using a different sample point $\mathbf{s}_2 = (0, 4, 0)$ classified to class c_2 . Since almost the whole feature space is already covered, the impact spaces are close *underapproximations* of both *class spaces*. Consequently, the void space between \mathbb{F}_{φ_1} and \mathbb{F}_{φ_2} is a guaranteed *overapproximation* of the *decision boundary*.

In Figure 9, we compare our explanations from Figure 8b with a state-of-the-art explanation [24] that cannot be expanded further feature-wise, using

ordering of the features (x_2, x_1, x_3) . Not only the space is much smaller, but the explanation also provides much less information with no feature relationships.