Formal Abductive Latent Explanations for Prototype-Based Networks

Jules Soria, Zakaria Chihani, Julien Girard-Satabin,
Alban Grastien, Romain Xu-Darme, Daniela Cancila
Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France
{jules.soria, zakaria.chihani, julien.girard2, alban.grastien, romain.xu-darme, daniela.cancila}@cea.fr

Abstract

Case-based reasoning networks are machine-learning models that make predictions based on similarity between the input and prototypical parts of training samples, called prototypes. Such models are able to explain each decision by pointing to the prototypes that contributed the most to the final outcome. As the explanation is a core part of the prediction, they are often qualified as "interpretable by design". While promising, we show that such explanations are sometimes misleading, which hampers their usefulness in safety-critical contexts. In particular, several instances may lead to different predictions and yet have the same explanation. Drawing inspiration from the field of formal eXplainable AI (FXAI), we propose Abductive Latent Explanations (ALEs), a formalism to express sufficient conditions on the intermediate (latent) representation of the instance that imply the prediction. Our approach combines the inherent interpretability of case-based reasoning models and the guarantees provided by formal XAI. We propose a solver-free and scalable algorithm for generating ALEs based on three distinct paradigms, compare them, and present the feasibility of our approach on diverse datasets for both standard and fine-grained image classification.

Code — https://github.com/julsoria/ale

1 Introduction

A widely adopted approach to explain neural network decisions is to analyze the decisions of a model after its training, in a post-hoc fashion (Molnar 2025). For neural networks in computer vision, a common line of work consists in computing the most relevant pixels by backpropagating gradients on the input space for a given sample (Smilkov et al. 2017; Sundararajan, Taly, and Yan 2017; Choi, Duplessis, and Belongie 2025; Montavon et al. 2019).

However, such approaches are not without flaws. They have been shown to be sensitive to malign manipulations (Dombrowski et al. 2019), raising questions on their usefulness in a setting where a stakeholder wants to provide correct explanations (Bordt et al. 2022). Moreover, some attribution methods may not correlate to the actual model behavior (Adebayo et al. 2018; Hedström et al. 2024) — raising questions on what they truly aim to explain — or display irrelevant features (Marques-Silva and Huang 2024). Finally, their usefulness on actual scenarios involving human users has been challenged (Colin et al. 2022).

To overcome such limitations, the emerging field of *formal explainable AI (FXAI)* (Audemard et al. 2022; Marques-Silva and Ignatiev 2022; Bassan and Katz 2023; Shi et al. 2020; Wolf, Galanti, and Hazan 2019) provides a rigorous framework to characterize and build explanations. A recent line of work (Marques-Silva and Ignatiev 2022; Bassan and Katz 2023; De Palma et al. 2025; Wu, Wu, and Barrett 2023) builds upon the framework of *abductive reasoning* where explanations are defined as a subset of input features that are sufficient to justify the model decision. In particular, it is possible to produce subset-optimal explanations within this framework, such that removing any single feature from such explanations can include elements that change the classifier's decision. FXAI provides strong guarantees on the relevance of features in the explanation, thanks to the use of automated provers that directly query the model. As a result, FXAI represents a good compromise between compactness and correctness — which are deemed important characteristics of an explanation (Nauta et al. 2023; Miller 2019).

Although FXAI is a promising approach, it suffers from two main shortcomings:

1. such approaches rely on expensive prover calls, impacting scalability on realistic computer vision tasks: such problems are generally NP-complete (Katz et al. 2017);
2. abductive FXAI provides explanations at the *feature-level*, which, for typical computer vision applications, is a pixel. We argue that the pixel-level is not the correct level of abstractions for the human final user of the explanation. Pixel-level explanations rely on the model's perception of the problem, creating a knowledge gap between the human and the machine (Miller 2019). On the other hand, prototypes or concepts allow generalizing facts towards higher-level reasoning (Lake et al. 2017).

Case-based reasoning is an orthogonal approach to FXAI to explain neural network decisions. Under this setting, the neural network is designed to justify its decision by exposing examples from its dataset that are similar to the new sample. Such approach is exemplified by prototype learning (Chen et al. 2019a; Rymarczyk et al. 2021; Van Looveren and Klaise 2021; Nauta, van Bree, and Seifert 2021; Rymarczyk et al. 2022; Willard et al. 2024; Sacha et al. 2024) or concept learning (Kim et al. 2018; Fel et al. 2023; Santis et al. 2025; Koh et al. 2020; Espinosa Zarlenga et al. 2022; Helbling et al. 2025). Prototype-based approaches have been shown to be human-interpretable through user-studies (Davoodi, Mohammadizadehsamakosh, and Komeili 2023).

Case-based reasoning explanations justify the decision by exposing a certain number of prototypes or concepts to the user. One major drawback is that the prototypes are usually not sufficient to entail the decision. In the original ProtoPNet (Chen et al. 2019b) architecture, the number of prototypes included in the explanation is fixed as an arbitrary hyperparameter. The resulting explanations could omit relevant prototypes, resulting in misleading explanations.

Contributions

In this paper, we aim to bridge the gap between FXAI and prototype-based learning. We show that, given a trained case-based reasoning network, we can produce abductive explanations that are correct not only at the pixel-level, but also in the latent space. We consider that the produced explanations are more suitable for humans than pixel-level ones, while being sufficient to justify the model's behavior.

We propose a framework to describe Abductive Latent Explanations (ALE) for prototype-based networks. Crucially, our formalism is generic as it can be instantiated given a definition of *feature extractor*, *prototype* and how *similarity* links prototypes to the current sample.

We provide three paradigms for computing ALEs, in order to circumvent the need for costly prover calls, and produce explanations that guarantee the prediction.

We evaluate our approach on numerous datasets and architectures variations to show the feasibility of building ALEs and their drawbacks.

2 Preliminaries

Our approach requires a system with the following components:

1. an **image encoder** f whose goal is to map the input from an input space $\mathcal{F}$ to a latent representation space $\mathcal{Z}$;
2. a **prototype layer** that measures the distance between the *latent* representation $\mathbf{z}$ of the image and learned prototypes $\mathbf{p}_j, j \in \mathbf{P}$, and assigns to them an *activation* score;
3. a **decision layer** $\kappa_{\mathcal{Z}}$ that leverages the prototype activation scores to perform instance classification.

In the rest of our paper, we leverage the prototype and decision layers as defined in the original ProtoPNet architecture (Chen et al. 2019b) and follow-ups (Willard et al. 2024) as they represent the state-of-the-art of prototype-based models, and implementations are readily available (Xu-Darme et al. 2024).

To give an intuition on abductive explanations in the latent space, we provide a running example after briefly introducing the notations used in our paper.

Notations

Refer to Table 1 for an overview of all notations. The predictor function $\kappa : \mathcal{F} \to \mathcal{C}$ is defined as $\kappa = \text{argmax} \circ h \circ \mathbf{a} \circ f$,

Symbol	Domain	Description
$\mathcal{C}$	Classes	Set of C classes, $\{1, \ldots, C\}$.
$\mathcal{F}$	$\mathbb{R}^{H_0 \times W_0 \times C_0}$	Input space (e.g., images).
$\mathcal{Z}$	$\mathbb{R}^{H_1 \times W_1 \times D}$	Latent representation space.
$\mathbb{P}$	$\mathbb{R}^{D}$	Latent component space.
$\mathbb{R}^m$		Prototype activation vector space.
$\mathbb{R}^C$		Class logit space.
$\mathbf{L}$	$\mathbb{N}$	component indices, e.g., $H_1 \times W_1$.
$\mathbf{P}$	$\mathbb{N}$	prototype indices, $\{1, \ldots, m\}$.
$\mathbf{v}; \mathbf{x}$	$\in \mathcal{F}$	Input instance (e.g., image).
$\mathbf{y}; \mathbf{z}$	$\in \mathcal{Z}$	Latent representation of $\mathbf{v}$; $\mathbf{x}$.
$\mathbf{z}_l$	$\in \mathbb{P}$	l-th latent component, $l \in \mathbf{L}$.
$\mathbf{p}_j$	$\in \mathbb{P}$	j-th prototype, $j \in \mathbf{P}$.
$\mathbf{a}(\mathbf{z})$	$\in \mathbb{R}^m$	Activation vector. j-th component is $\mathbf{a}_j(\mathbf{z})$.
c	$\in \mathcal{C}$	Predicted class. $c = \kappa(\mathbf{v})$.
m	$\in \mathbb{N}$	Number of prototypes.
$\mathcal{E}$	$\mathbf{P} \times \mathbf{L}$	Abductive Latent Explanation.
$\mathbf{a}_{\mathcal{E}}$	$\subseteq \mathbb{R}^m$	Constrained activation space.
$\overline{\mathbf{a}}_{\mathcal{E},j}$	$\in \mathbb{R}$	Upper bound for j-th activation.
$\underline{\mathbf{a}}_{\mathcal{E},j}$	$\in \mathbb{R}$	Lower bound for j-th activation.
$d(\cdot,\cdot)$	$\mathbb{P} \times \mathbb{P} \to \mathbb{R}$	Distance metric in $\mathbb{P}$.
$\sigma(\cdot)$	$\mathbb{R} \to \mathbb{R}$	Distance-to-similarity function.
$\text{sim}(\cdot,\cdot)$	$\mathbb{P} \times \mathbb{P} \to \mathbb{R}$	Similarity, $\text{sim}(a,b) = \sigma(d(a,b))$.
$\overline{\text{sim}}_{\mathcal{E}}(\cdot,\cdot)$	$\mathbb{P} \times \mathbb{P} \to \mathbb{R}$	Upper bound on similarity.
$\underline{\text{sim}}_{\mathcal{E}}(\cdot,\cdot)$	$\mathbb{P} \times \mathbb{P} \to \mathbb{R}$	Lower bound on similarity.
f	$\mathcal{F} \to \mathcal{Z}$	Maps inputs to latent space.
$\mathbf{a}$	$\mathcal{Z} \to \mathbb{R}^m$	Prototype activation function: $j \in \mathbf{P}.\mathbf{a}_j(\mathbf{z}) = \max_{l \in \mathbf{L}} \text{sim}(\mathbf{z}_l, \mathbf{p}_j)$.
h	$\mathbb{R}^m \to \mathbb{R}^C$	Outputs class logits.
κ	$\mathcal{F} \to \mathcal{C}$	Predictor function: $\kappa = \text{argmax} \circ h \circ \mathbf{a} \circ f$.
$\kappa_{\mathcal{Z}}$	$\mathcal{Z} \to \mathcal{C}$	Latent predictor function: $\kappa_{\mathcal{Z}} = \text{argmax} \circ h \circ \mathbf{a}$.

Table 1: Summary of Notations Used.

where f maps inputs to the latent space, $\mathbf{a}$ computes prototype activations, and h produces class logits. The similarity matrix $\mathbf{sim}(\mathbf{z}, \mathbf{p})$, with $\mathbf{p} = (\mathbf{p}_j)_{j \in \mathbf{P}}$, has entries $\text{sim}(\mathbf{z}_l, \mathbf{p}_j)$ for $l \in \mathbf{L}$, $j \in \mathbf{P}$. Explanations $\mathcal{E}$ are a subset of the latent features in $\mathbf{P} \times \mathbf{L}$.

Running example

We consider a 2-class classification problem with `emperor_penguin` and `royal_penguin`. During inference, an image input $\mathbf{v}$ first goes through an **image encoder** f and is represented as $\mathbf{z}$, i.e. an object in the latent space $\mathcal{Z}$, which sums up the important concepts *locally* identified. This latent representation is composed of 4 ($H_1 \times W_1$) latent vectors, with width $W_1 = 2$ and height $H_1 = 2$ (Figure 1).

A *prototype* is a latent vector extracted from a training image that the training procedure has identified as representative of some class. In our example, we assume that the training procedure computed five prototypes, akin to the following concepts: `emperor_beak`, `emperor_yellow_neck_patch`, `black_back_`

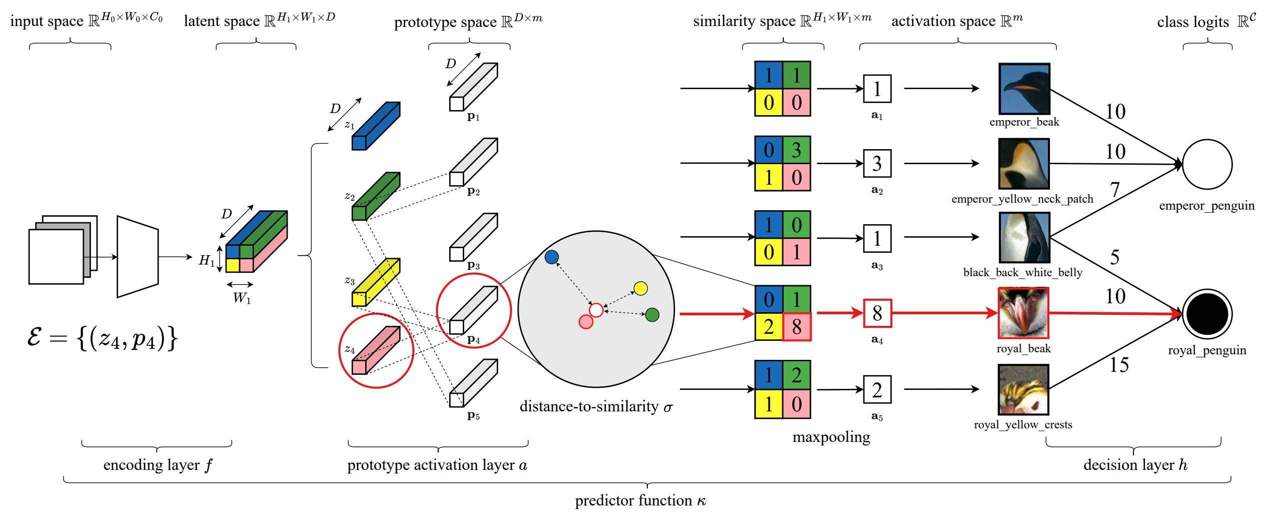
input space $\mathbb{R}^{H_0 \times W_0 \times C_0}$
latent space $\mathbb{R}^{H_1 \times W_1 \times D}$
prototype space $\mathbb{R}^{D \times m}$
similarity space $\mathbb{R}^{H_1 \times W_1 \times m}$
activation space $\mathbb{R}^m$
class logits $\mathbb{R}^{\mathcal{C}}$
D
H_1
W_1
z_1
z_2
z_3
z_4
$\mathbf{p}_1$
$\mathbf{p}_2$
$\mathbf{p}_3$
$\mathbf{p}_4$
$\mathbf{p}_5$
$\mathcal{E} = \{(z_4, p_4)\}$
distance-to-similarity σ
$\mathbf{a}_1$
$\mathbf{a}_2$
$\mathbf{a}_3$
$\mathbf{a}_4$
$\mathbf{a}_5$
maxpooling
emperor_beak
emperor_yellow_neck_patch
black_back_white_belly
royal_beak
royal_yellow_crests
emperor_penguin
royal_penguin
encoding layer f
prototype activation layer a
decision layer h
predictor function κ

Figure 1: Example of a top-1 explanation for a ProtoPNet with five prototypical parts for two classes.

white_belly, royal_beak, and royal_yellow_crests (these labels are solely used to make the example easy to follow; they are neither inputs of the training nor produced by it). We use $\mathbf{p}$ to refer to the vector of prototypes and $\mathbf{p}_j$ for the j-th prototype.

Secondly, the **prototype layer** $\text{sim}(\mathbf{z}, \mathbf{p})$ computes a similarity score between each latent vector (row) and the prototypes (column), where higher value means higher similarity:

$$\text{sim}(\mathbf{z}, \mathbf{p}) = \begin{bmatrix} 1. & 0. & 1. & 0. & 1. \\ 1. & 3. & 0. & 1. & 2. \\ 0. & 1. & 0. & 2. & 1. \\ 0. & 0. & 1. & 8. & 0. \end{bmatrix}$$

In this example, the fourth prototype (royal_beak, fourth column) has been strongly recognized in the bottom-right part of the image (fourth row), hence a similarity score of 8. Similarly, the second prototype (emperor_yellow_neck_patch) is moderately recognized, with a score of 3., while the other prototypes are absent (score 2 or lower). For each prototype, the ProtoPNet architecture is only interested in the top similarity, and computes the column-wise maximum value of $\text{sim}(\mathbf{z}, \mathbf{p})$ called the *activation vector*

$$\mathbf{a}(\mathbf{z}) = [1. \quad 3. \quad 1. \quad 8. \quad 2.]$$

These values mean that the network detected a clear royal beak and what appears to be an emperor yellow neck patch, but no other attribute one would expect from a penguin.

Thirdly, in the **decision layer**, the *activation* vector is fed to a fully connected linear layer with learned parameters W:

$$W = \begin{bmatrix} 10. & 10. & 7. & 0. & 0. \\ 0. & 0. & 5. & 10. & 15. \end{bmatrix}$$

Matrix W indicates that prototypes 1 and 2 are typical of emperor penguins (weights 10. against 0.) and prototypes 4 and 5 of royal penguins (weights 10. and 15. versus 0.), while prototype 3 can be detected in both classes. The decision layer's computation returns the final class scores:

$$h(\mathbf{a}(\mathbf{z})) = W\mathbf{a}(\mathbf{z}) = [47. \quad 115.]$$

and the classifier thus outputs the second class: $\kappa_{\mathcal{Z}}(\mathbf{z}) = \text{argmax}\, h(\mathbf{a}(\mathbf{z})) = 2$, i.e., royal_penguin.

ProtoPNet returns as an explanation the k prototypes with highest activation. In the running example, if $k = 1$, the explanation is $\mathcal{E} = \{(\mathbf{z}_4, \mathbf{p}_4)\}$ which involves $\text{sim}(\mathbf{z}_4, \mathbf{p}_4) = 8$. We highlight that the above explanation implicitly entails that the score of all other prototypes is 8 or below. At first glance, such an explanation can appear appropriate: the bird on the image is classified as a royal_penguin because a royal beak (typical of royal penguins) has been observed.

However, a closer examination reveals that the explanation is wrong for some instances. Let us consider the following counter-example: an image having latent representation $\mathbf{z}'$ such that the activation evaluates to

$$\mathbf{a}(\mathbf{z}') = [6. \quad 7. \quad 1. \quad 8. \quad 2.]$$

The classification is $\kappa_{\mathcal{Z}}(\mathbf{z}') = \text{argmax}\,[137. \quad 115.] = 1$ while the previous explanation ("*because it shows a royal beak*") is still applicable to emperor_penguin. This stems from the fact that, following ProtoPNet fashion, we defined the explanation to only take the *top-activated prototype*, while other, less active but still important prototypes, are needed to fully justify the decision. In other words, the explanation is misleading or *optimistic* as discussed in (Ignatiev, Narodytska, and Marques-Silva 2019). Hence, the need for formal guarantees on prototypes explanations — abductive latent-based explanations.

3 Abductive Latent Explanations

An Abductive eXplanation (AXp) for $\mathbf{v}$ is traditionally defined as a condition on the input of a classifier satisfied by the current instance such that all instances that satisfy this condition yield the same output (Ignatiev, Narodytska, and Marques-Silva 2019). In other words, an AXp defines *preconditions* on the inputs of a classifier, yielding a *postcondition* on the classification of said classifier (Dijkstra 1976). Formally, using notations presented in Table 1, given an instance $\mathbf{v}$ with prediction c, an *abductive explanation* is a precondition $\phi_{\mathcal{E}}(\mathbf{x}, \mathbf{v})$ over input $\mathbf{x}$ satisfied by $\mathbf{v}$ such that:

$$\forall \mathbf{x} \in \mathcal{F}. \quad \phi_{\mathcal{E}}(\mathbf{x}, \mathbf{v}) \Rightarrow (\kappa(\mathbf{x}) = c).$$

In previous works (Marques-Silva and Ignatiev 2022), the explanation is represented as a subset $\mathcal{E}$ of input variables (features), and the precondition $\phi_{\mathcal{E}}$ simply states that the assignments of these variables should match those of $\mathbf{v}$:

$$\phi_{\mathcal{E}}(\mathbf{x}, \mathbf{v}) = \bigwedge_{i \in \mathcal{E}} (\mathbf{x}_i = \mathbf{v}_i).$$

Our main contribution is the extension of the definition of AXps to an *arbitrary latent space*.

In the case of image recognition, an explanation is then a subset of the image pixels. While this explanation is correct (i.e., any image that includes these specific pixels will be classified the same) its interpretability is questionable.

Since explanations are preconditions on the input space of the classifier, in the context of case-based reasoning, we can define such preconditions on the *input of the latent classifier*.

Definition 1 (Abductive Latent Explanation (ALE)). *Given an input instance* $\mathbf{v}$ *with latent representation* $f(\mathbf{v})$*, an* abductive latent explanation *is a subset of latent features* $\mathcal{E}$ *that entails a precondition* $\phi_{\mathcal{E}}$ *over* $f(\mathbf{x})$ *satisfied by* $f(\mathbf{v})$*.*

Given an input space $\mathcal{F}$ *and a predictor* $\kappa = \kappa_{\mathcal{Z}} \circ f$ *from* $\mathcal{F}$ *to* $\mathcal{C}$*, the explanation is* correct *if the following holds:*

$$\forall \mathbf{x} \in \mathcal{F}. \quad \phi_{\mathcal{E}}(f(\mathbf{x}), f(\mathbf{v})) \Rightarrow (\kappa(\mathbf{x}) = c).$$

If $\mathcal{E}$ is an ALE, it is not necessarily subset-minimal. Indeed, when $\mathcal{E} \subseteq \mathbf{P} \times \mathbf{L}$ then $\mathbf{P} \times \mathbf{L}$ is an ALE. We take a particular interest in defining (and later computing) subset-minimal explanations, assuming that a smaller subset is more human-interpretable than the whole superset.

Definition 2 (Subset-Minimal ALE). *An Abductive Latent Explanation* $\mathcal{E}$ *for an instance* $\mathbf{v}$ *is* ***subset-minimal*** *if no proper subset of it is also an ALE for the same instance. This can be expressed formally as:*

$$\forall \mathcal{E}' \subsetneq \mathcal{E}, \;\; \exists \mathbf{x}' \in \mathcal{F} \text{ s.t. } \Big(\phi_{\mathcal{E}'}(f(\mathbf{x}'), f(\mathbf{v})) \wedge (\kappa(\mathbf{x}') \neq c)\Big).$$

Similarly, we can formalize the implicit definition of an explanation that is used in ProtoPNet as follows:

Definition 3 (ProtoPNet Explanation). *Given a set* $\mathbf{P}$ *of prototype indices, a* ProtoPNet explanation $\mathcal{E} \subseteq \mathbf{P}$ *is a subset of indices that implicitly represents the precondition* $\phi_{\mathcal{E}}$*:*

$$\phi_{\mathcal{E}}(\mathbf{z}, \mathbf{y}) = \left(\bigwedge_{i \in \mathcal{E}} \mathbf{a}_i(\mathbf{z}) = \mathbf{a}_i(\mathbf{y}) \right) \wedge \left(\bigwedge_{\substack{j \notin \mathcal{E} \\ i \in \mathcal{E}}} \mathbf{a}_j(\mathbf{z}) \leq \mathbf{a}_i(\mathbf{y}) \right).$$

Compared to the pixel-based abductive explanations, we consider that ALEs are more interpretable as they refer to the concepts that humans are able to manipulate. Furthermore, compared to relevance-based explanations, ALEs provide formal guarantees as it is impossible to come up with misleading (or optimistic) explanations (i.e., no sample matching the explanation would be classified differently).

4 Building abductive explanations in the latent space

In this section, we show how to design the precondition entailed by an ALE that relies on the bounds of intermediate prototype activation scores. Thus, an explanation $\mathcal{E}$ will implicitly entail a condition of the form

$$\phi_{\mathcal{E}}(\mathbf{z}, \mathbf{y}) = \left(\wedge_{j=1}^{m} \mathbf{a}_j(\mathbf{z}) \in \left[\underline{\mathbf{a}}_{\mathcal{E},j}, \overline{\mathbf{a}}_{\mathcal{E},j} \right] \right).$$

In this setting, recall from Section 2 that activations are computed from similarity scores, such that $\mathrm{sim}(a, b) = \sigma(d(a, b))$ where σ is monotonous function which *increases* as the distance *decreases*[1]. Boundaries on one can be translated to boundaries on the other. Therefore, we focus next on deducing boundaries on the distance between prototypes and feature vectors, which propagate into boundaries in the activation space, then into boundaries in the class logits space.

Domain and Spatial Constraints

Domain-restricted AXps (Yu et al. 2023) refine the classical definition by constraining ”unfixed” variables in the domain space. Similarly, we note that prototypes are fixed in the latent space $\mathcal{Z}$. As such, latent space feature vectors are bounded by their distance to these prototypes. Thus, we can define a precondition which relies on the *relationship between a latent vector and its location* w.r.t. *other prototypes*.

A cornerstone of our approach is to analyse how giving more information about the distance between a latent component and a prototype changes the interval boundaries for the *other prototype activation scores*, exploiting these spatial constraints. Furthermore, what follows in this section is true for any *true* distance function, such as the L_2 distance.

We propose two methods to compute the interpretation (the bounds on the activation) of these explanations. In our case, an ALE $\mathcal{E}$ is a subset of $\mathbf{P} \times \mathbf{L}$ (where $\mathbf{L}$ is a set of indices of components in $\mathcal{Z}$, as defined in Table 1).

Triangular Inequality In order to reach prototype activation score boundaries while ensuring stronger “awareness” that the prototypes and feature vectors co-exist in the same dimensional space $\mathbb{P}$, we propose to use the *triangular inequality*, a property of distance functions. Given $\mathbf{z}_l, \mathbf{p}_i, \mathbf{p}_j$:

$$|d(\mathbf{z}_l, \mathbf{p}_j) - d(\mathbf{p}_j, \mathbf{p}_i)| \leq d(\mathbf{z}_l, \mathbf{p}_i) \leq d(\mathbf{z}_l, \mathbf{p}_j) + d(\mathbf{p}_j, \mathbf{p}_i).$$

By definition, the prototypes $\mathbf{p}_{1,\ldots,m}$ are an integral part of the classifier model; we have access to $d(\mathbf{p}_i, \mathbf{p}_j), \forall i, j$.

Given the distance between a singular feature vector $\mathbf{z}_l$ and a prototype $\mathbf{p}_j$, the Triangular Inequality naturally provides boundaries on the similarity score $\mathrm{sim}(\mathbf{z}_l, \mathbf{p}_i)$ between

[1]In (Chen et al. 2019b), the function $\sigma(x) = \log\left(\frac{x+1}{x+\epsilon}\right)$ is used.

that feature vector $\mathbf{z}_l$ and all other prototypes $\mathbf{p}_{i \neq j}$. From these similarity boundaries (for all feature vector-prototype pairs), an upper and lower bound on the prototype activation score $\mathbf{a}_\mathcal{E}$ can be deduced. An ALE $\mathcal{E}$ then entails $\phi_\mathcal{E}$ with the boundaries $\{\underline{\text{sim}}_\mathcal{E}, \overline{\text{sim}}_\mathcal{E}\}$ defined through:

$$\forall (l,j) \in \mathcal{E}.\ \overline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j) = \underline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j) = \text{sim}(\mathbf{z}_l, \mathbf{p}_j)$$
$$\forall (l,i) \notin \mathcal{E}.\ \underline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_i) = \max_{(l,j)\in\mathcal{E}} \sigma(d(\mathbf{z}_l, \mathbf{p}_j) + d(\mathbf{p}_j, \mathbf{p}_i))$$
$$\overline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_i) = \min_{(l,j)\in\mathcal{E}} \sigma(|d(\mathbf{z}_l, \mathbf{p}_j) - d(\mathbf{p}_j, \mathbf{p}_i)|).$$

These boundaries give $\mathbf{a}_\mathcal{E}$ obtained by,

$$\forall j \in \mathbf{P}.\quad \underline{\mathbf{a}}_{\mathcal{E},j} = \max_{l\in\mathbf{L}} \underline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j)$$
$$\overline{\mathbf{a}}_{\mathcal{E},j} = \max_{l\in\mathbf{L}} \overline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j).$$

Hypersphere Intersection Approximation An alternative way of devising such constraints is to see the latent feature vectors as intersections between hyperspheres - where those hyperspheres have the prototypes as centers, and the distance between the vector and those prototypes as radii. It is thus possible to deduce distance boundaries by *overapproximating hyperspheres intersection*. Such intersection is itself a hypersphere that contains the latent feature vector. In this spatial paradigm, at each "iteration", i.e., for a latent feature vector, every time we update/extend the explanation by adding to it a distance between that vector and a new prototype, we refine the size of the hypersphere containing it.

One advantage of this approximation is that the resulting hypersphere is, at worst, the same size (radius-wise) as the smallest of the two intersecting hyperspheres. When we use the previous approximation as one of the two hyperspheres, we have the guarantee that the next approximation will necessarily be better. In contrast, when relying on the triangular inequality, adding a feature-prototype pair to the explanation may not help refine the boundaries concerning that feature vector. An additional figure which helps visualise the approximation can be found in the Appendix.

Definition 4 (Hypersphere Intersection Approximation)**.** *Let H_1 and H_2 be two hyperspheres in a Euclidean space $\mathbb{R}^D$ with centers $\mathbf{C}_1, \mathbf{C}_2 \in \mathbb{R}^D$ and radii $r_1, r_2 > 0$, respectively. Assume the hyperspheres have a non-empty intersection, which requires the distance between centers $d = \|\mathbf{C}_1 - \mathbf{C}_2\|_2$ to satisfy $|r_1 - r_2| \leq d \leq r_1 + r_2$. Let H_3 be the hypersphere approximating the intersection $H_1 \cap H_2$, constructed as described above. Then, H_3 has a radius r_3:*

$$r_3 = \frac{2}{d}\sqrt{p(p-d)(p-r_1)(p-r_2)}$$

where $p = \frac{1}{2}(d + r_1 + r_2)$. The center $\mathbf{C}_3$ of H_3 is:

$$\mathbf{C}_3 = \mathbf{C}_1 + \sqrt{r_1^2 - r_3^2} \cdot \frac{\mathbf{C}_2 - \mathbf{C}_1}{d}.$$

Theorem 1 (Containment and Minimality of the Hypersphere Intersection Approximation)**.** *Let H_1 and H_2 be two hyperspheres in a Euclidean space $\mathbb{R}^D$ with centers $\mathbf{C}_1, \mathbf{C}_2$ and radii r_1, r_2, respectively. Let their surfaces intersect in a non-empty set S_{int}, defined as:*

$$S_{int} = \{\mathbf{s} \in \mathbb{R}^D \mid \|\mathbf{s} - \mathbf{C}_1\|_2 = r_1 \text{ and } \|\mathbf{s} - \mathbf{C}_2\|_2 = r_2\}.$$

Let H_3 be the approximating hypersphere with center $\mathbf{C}_3$ and radius r_3 as defined in Definition 4. Then:

1. ***(Containment)** The intersection of the surfaces is entirely contained within the approximating hypersphere H_3. That is, $S_{int} \subseteq H_3$.*
2. ***(Minimality)** H_3 is the smallest possible hypersphere by radius that contains S_{int}. For any hypersphere H' with radius r' such that $S_{int} \subseteq H'$, it must be that $r_3 \leq r'$.*

Proof. Found in the Appendix. □

With this paradigm, an ALE $\mathcal{E}$ entails $\phi_\mathcal{E}$ with $\mathbf{a}_\mathcal{E}$:

$$\forall (l,j) \in \mathcal{E}.$$
$$\overline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j) = \underline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j) = \text{sim}(\mathbf{z}_l, \mathbf{p}_j)$$
$$\forall (l,j) \notin \mathcal{E}.$$
$$\overline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j) = \sigma(d(C_{\mathbf{z}_l}, \mathbf{p}_j) - r_{\mathbf{z}_l})$$
$$\underline{\text{sim}}_\mathcal{E}(\mathbf{z}_l, \mathbf{p}_j) = \sigma(d(C_{\mathbf{z}_l}, \mathbf{p}_j) + r_{\mathbf{z}_l})$$

with $C_{\mathbf{z}_l}$ and $r_{\mathbf{z}_l}$ the center and radius, respectively, of the approximated hypersphere containing $\mathbf{z}_l$.

This approximation is possible because, by construction, all considered hyperspheres **do** intersect in at least one point, the latent feature vector we attempt to approximate.

Top-k explanations This paradigm is the one used implicitly by the original explanation (Chen et al. 2019b) provided by ProtoPNet. It traverses the prototype activation scores in decreasing order, with the added knowledge that, for that prototype, the similarity scores with the other latent space feature vectors are lesser than the activation score (result of the `max` function). In that scenario, we obtain $\mathbf{a}_\mathcal{E}$ from $\mathcal{E}$ by:

$$\forall j \in \mathcal{E}.\quad \underline{\mathbf{a}}_{\mathcal{E},j} = \overline{\mathbf{a}}_{\mathcal{E},j} = \mathbf{a}_j(\mathbf{y})$$
$$\forall j \notin \mathcal{E}.\quad \underline{\mathbf{a}}_{\mathcal{E},j} = 0 \ \text{ and } \ \overline{\mathbf{a}}_{\mathcal{E},j} = \min_{i\in\mathcal{E}} \mathbf{a}_i(\mathbf{y}).$$

Notice that, in the top-k explanation, the explanation is based on the activation space directly. The latent feature vector relevant for that activation is implicitly included in $\mathcal{E}$ so it behaves as if $\|\mathbf{L}\| = 1$.

Constructing Abductive Latent Explanations

Given a candidate explanation $\mathcal{E} \subseteq \mathbf{P} \times \mathbf{L}$, the constrained **prototype activation** space is represented by

$$\mathbf{a}_\mathcal{E} = \{[0,\ \overline{\mathbf{a}}_{\mathcal{E},j}]\}_{j\in\mathbf{P}}.$$

Following our preliminary work (Soria et al. 2025), we introduce the notion of constructing an abductive latent explanation $\mathcal{E}$ in the prototype activation space by defining preconditions on the activation vector $\mathbf{a} \in \mathbb{R}^m$. These preconditions are derived from $\mathcal{E}$ and $\mathbf{a}(\mathbf{y})$. Crucially, they must guarantee that any activation vector $\mathbf{a}$ satisfying these conditions results in the same predicted class c:

$$\forall \mathbf{a} \in \mathbf{a}_\mathcal{E} \subseteq \mathbb{R}^m : \mathbf{a} \models \phi_\mathcal{E}, \quad \underset{k\in\mathcal{C}}{\text{argmax}}\, h_k(\mathbf{a}) = c.$$

These preconditions effectively define a constrained region - or set of constraints $\mathbf{a}_\mathcal{E}$ within the activation space $\mathbb{R}^m$ - such that $\mathbf{a}(\mathbf{y})$ satisfies these constraints, and all vectors within this region also yield the prediction c.

Assumption 1 (Linear Logit Difference)**.** *For any two classes k, c, the difference between their logit functions is linear in the prototype activation vector* $\mathbf{a}$*:*

$$h_k(\mathbf{a}) - h_c(\mathbf{a}) = \sum_{j=1}^{m}(w_{jk} - w_{jc})\,\mathbf{a}_j + (b_k - b_c)$$

for some weights w_{jk}, w_{jc} and biases b_k, b_c.

Definition 5 (Maximally Class-Favoring Element within $\mathbf{a}_{\mathcal{E}}$)**.** *For a given explanation $\mathcal{E}$, predicted class c, and class $k \neq c$, the* maximally (k/c)-favoring **prototype activation** vector within $\mathbf{a}_{\mathcal{E}}$ *is denoted by $\mathbf{a}^*_{\mathcal{E}}(k,c)$ and satisfies:*

$$\forall \mathbf{a} \in \mathbf{a}_{\mathcal{E}}.\quad h_k(\mathbf{a}^*_{\mathcal{E}}(k,c)) - h_c(\mathbf{a}^*_{\mathcal{E}}(k,c)) \geq h_k(\mathbf{a}) - h_c(\mathbf{a}).$$

Under Assumption 1, its components $(\mathbf{a}^*_{\mathcal{E}}(k,c))_j$ are constructed as:

$$(\mathbf{a}^*_{\mathcal{E}}(k,c))_j = \begin{cases} (\overline{\mathbf{a}}_{\mathcal{E}})_j & \textit{if}\ w_{jk} \geq w_{jc} \\ (\underline{\mathbf{a}}_{\mathcal{E}})_j & \textit{if}\ w_{jk} < w_{jc} \end{cases}.$$

Intuitively, this element *satisfies* the condition expressed by the explanation $\mathcal{E}$, and prevents the most the classifier from reaching its initial prediction c (in favor of class k).

Definition 6 (Class-wise Prediction Domination within $\mathbf{a}_{\mathcal{E}}$)**.** *For explanation $\mathcal{E}$, predicted class c, and alternative class $k \neq c$, we say c* dominates *k within $S_{\mathcal{E}}$, denoted $\psi_{\mathcal{E}}(k,c)$, if the logit of c is greater than or equal to the logit of k even for the maximally (k/c)-favoring vector:*

$$\psi_{\mathcal{E}}(k,c) \iff h_c(\mathbf{a}^*_{\mathcal{E}}(k,c)) \geq h_k(\mathbf{a}^*_{\mathcal{E}}(k,c)).$$

Definition 7 (Total Prediction Domination within $\mathbf{a}_{\mathcal{E}}$ (Explanation Verification))**.** *A candidate explanation $\mathcal{E}$ is considered* verified *if class c dominates all other classes $k \neq c$ within the constrained space $S_{\mathcal{E}}$:*

$$\textit{Verify}(\mathcal{E}) \iff \forall k \in \{1, \ldots, C\} \setminus \{c\}, \quad \psi_{\mathcal{E}}(k,c).$$

In Definition 6 we say that a class (different from the predicted class) is *verified* if its *Maximally Class-Favoring Element* does not entail a different classification. In Definition 7 we say that the explanation *verifies* the prediction if **all** classes are *verified*.

If an explanation $\mathcal{E}$ is verified according to Definition 7, it is an abductive explanation as presented in Definition 1.

Theorem 2 (Verified Explanation Sufficiency)**.** *Let $\mathcal{E}$ be a candidate explanation for the decision $(\mathbf{v}, c)$. If $\mathcal{E}$ is verified according to Definition 7 (i.e., Verify$(\mathcal{E})$ is true), then $\mathcal{E}$ is a valid abductive explanation according to Definition 1.*

Proof. Found in the Appendix. □

Algorithms

We present two algorithms for generating ALEs of model predictions. Pseudo-code for both algorithms can be found in the Appendix. Algorithm 1 relies on the top-k paradigm, where we iteratively add the next highest activated prototype to the running explanation, until the candidate is verified. NEXTPROTOTYPE selects a prototype from $\mathcal{A}$ the ordered list of prototypes by activations. Due to the fixed order of elements, the produced explanation is **sufficient** and **cardinality-minimal**.

Algorithm 2 relies on spatial constraints, as defined earlier in this section. We first go through a *forward pass* where we iteratively add pairs of latent components and prototypes. From this candidate explanation, we deduce bounds on the activation space, then verify whether the running explanation's bounds guarantee the prediction. At the end of this stage, the candidate explanation is **sufficient**. Carefully, we add a *backward pass* which downsizes the candidate explanation, i.e., iteratively attempts to remove pairs, keeping only those that lead the resulting pruned explanation to not be verified anymore, until no pair can be removed — this provides the **subset-minimality** property. While the search is *quadratic* in the explanation's length, this complexity remains polynomial. This is a significant advantage over traditional formal methods (e.g., SMT/MILP-based) for pixel-wise explanations, which are *NP-hard*.

Most importantly, these two algorithms do not rely on calling any external solvers.

5 Experimental Study

Computational Resources Our experiments were conducted on a SLURM-managed computing cluster, primarily utilizing GPU-equipped nodes for computationally intensive tasks, with each node featuring 48 cores at 2.6GHz, 187GB of RAM, and four NVIDIA V100 32GB GPUs.

Methodology Our study leverages the Case-Based Reasoning Network (CaBRNet) framework (Xu-Darme et al. 2024) to train different models. We trained one ProtoPNet model per dataset considered. For the backbone, depending on the dataset resolution and number of classes, we used either a VGG (Simonyan and Zisserman 2015), a ResNet (He et al. 2016), or a WideResNet (Zagoruyko and Komodakis 2016), using either the 'default' ImageNet (Russakovsky et al. 2015) pretrained weights, or the iNaturalist (Su and Maji 2021) pretrained weights. We parameterized our models such that 10 prototypes per class were initially allocated (as is the standard for case-based reasoning training).

Existing Python libraries are known to exhibit non-trivial errors when computing distances.[2] This leads us to (sometimes) over-approximate bounds, which may also affect the cardinality of the explanations.

Datasets In order to compute *subset-minimal* ALEs, we trained several ProtoPNet models on a variety of datasets, including CUB200 (Wah et al. 2011), Stanford Cars (Krause et al. 2013), Oxford Pet (Parkhi et al. 2012), Oxford Flowers (Nilsback and Zisserman 2008), CIFAR-10 and CIFAR-100 (Krizhevsky, Hinton et al. 2009), and MNIST (LeCun 1998). These datasets were chosen for their diversity and relevance to the task at hand. Several of these datasets exhibit a hierarchical structure that allows them to be used for both fine-grained and coarse-grained classification tasks.

When possible, we used the standard train/test splits provided by the dataset creators and followed common preprocessing procedures, such as resizing, cropping, and data augmentation techniques, described in (Chen et al. 2019b).

[2]cf. issue https://github.com/pytorch/pytorch/issues/57690

Dataset	*Accuracy*	Triangle	Hypersphere	top-k	$H_1 \times W_1$	top-k (adj.)
		Avg Total / Avg Correct / Avg Incorrect				
CIFAR-10	0.83	**8.7 / 6.6 / 19.4**	20.2 / 8.9 / 28.8	41.4 / 36.9 / 61.9	1×1	41.4 / 36.9 / 61.9
CIFAR-100†	0.62	**323.2 / 276.7 / 394.3**	672.9 / 574.4 / 820.2	896.6 / 867.6 / 940.8	1×1	672.9 / 867.6 / 940.8
MNIST†	0.98	**6.2 / 6.2** / -	675 / 675 / -	8.8 / 8.8 / -	4×4	141 / 141 / -
Oxford Flowers†	0.72	546.9 / 394.8 / 973.5		**287.7 / 193.6 / 525.5**	4×4	4602 / 3098 / 8408
Oxford Pet†	0.82	3755.3 / 748.9 / 18130		**77.7 / 67.9 / 122.8**	7×7	3805 / 3328 / 6016
Stanford Cars†	0.90	5072.3 / 992.1 / 31633.6		**24.9 / 12.3 / 140.6**	7×7	1219 / 600 / 6890
CUB200†	0.84	10653.4 / 670.9 / 98000		**239.3 / 217.0 / 352.0**	7×7	11725 / 10632 / 17251

Table 2: Summary of Average Explanation Sizes, Correct/Incorrect Explanation Sizes, for each dataset. Accuracy on the datasets is given only for reference. The latent space dimensions are also referenced to show how the top-k explanation size is 'adjusted' to fairly compare with spatial constraints ALEs. † indicates that for that dataset, the spatial constraints ALEs are computed on five randomly selected images per class. For MNIST, since no incorrect prediction was sampled, there are no obtained results on incorrect explanations sizes. For some higher resolution datasets, generating Hypersphere Intersection Approximation ALEs proved to be too computation intensive, and exceeded the two-day timeout. Best (smallest) results indicated in bold.

Training models on these datasets was facilitated by their implementation in the `torchvision` library (maintainers and contributors 2016).

Results Table 2 summarizes the average sizes of explanations on all datasets, broken down by correct and incorrect classifications, for each paradigm. When looking at the computations for top-k, we show that the 10 most activated prototypes are usually not enough to guarantee the decision. On all datasets except MNIST, explanations on average require more than 10 similarity scores to be sufficient to justify the classification. This result is significant as it entails that, for the models used, nearly all original ProtoPNet explanations (according to Definition 3) are misleading, or optimistic.

Furthermore, we observe in Table 2 that, for samples that are incorrectly classified the average explanation size is much higher. This would mean that information about a sample and the data distribution can be extracted from the ALEs themselves. This wide disparity is also linked to the computation time to generate an ALE — as the candidate explanation grows during the forward pass, the time it takes to append an additional latent vector-prototype pair grows too. The observed phenomenon is exacerbated for spatial constraints ALE; for the CUB200, the Oxford Pet and Oxford Flowers datasets, incorrect predictions require an ALE that includes all possible pairs (i.e., $\mathcal{E} = \mathbf{P} \times \mathbf{L}$). One interpretation of such result is that ALEs are relevant to understand the prediction only when it is correct. This finding corroborates with related work (Wu et al. 2024) which uses the explanation size as proxy for out-of-distribution detection.

Another observation from our experiments is that, for small resolution datasets, the spatial constraints ALEs (for correct predictions) require *less* pairs than top-k ALEs, even before accounting for the relevant adjustments (multiplying by the number of latent components). We can deduce that, in certain contexts at least, formally proving the classification of an instance using our developed paradigms leads to more compact explanations.

Discussion We show that producing subset-minimal and sufficient explanations leads to explanations of large size, and therefore arguably not human-interpretable. This is not a limitation of our method but rather a key finding: current prototype-based networks need a lot of components to yield explanations with formal guarantees. This discovery nuances their previous claim of interpretability.

6 Conclusion

In this work, we introduced Abductive Latent Explanations (ALE), a novel framework that formalizes explanations for prototype-based networks as rigorous abductive inferences within the latent space. We proposed a solver-free algorithm for generating ALEs that drastically reduces computation time. Our analysis, enabled by ALEs, reveals that common prototype-based explanations can be misleading, raising concerns about their reliability in high-stakes decisions. Furthermore, we investigated the relationship between prediction correctness and ALE size, with findings showing that larger ALEs often correlate with incorrect predictions, suggesting ALE size as a potential proxy for model uncertainty.

Limitations and Future Works

A current limitation of our approach is the size of the obtained explanations. Explanations in the order of magnitude of thousands for a single instance can arguably be considered still too big to be actionable and interpretable by humans. Furthermore, prototypes are currently not directly linked to human concepts. We consider bridging symbolic reasoning and ALEs a worthwile and interesting research direction. Finally, exploring training methods which lead to a more interpretable latent space, such as well separated class prototypes (Chen et al. 2019b) and meaningful encodings (Chen, Bei, and Rudin 2020) could help in generating more compact and valuable ALEs.

A direct extension of our work consists on taking further inspirations from Abductive Explanations to generate Constrastive Explanations. Extending ALEs to other prototypes modalities is also a potential future work. ALEs could also be useful to identify irrelevant latent components, opening new perspective on targeted pruning for neural networks and their explanations.

Acknowledgements

This publication was made possible by the use of the FactoryIA supercomputer, financially supported by the Ile-De-France Regional Council. This work was supported by the SAIF project, funded by the "France 2030" government investment plan managed by the French National Research Agency, under the reference ANR-23-PEIA-0006

References

Adebayo, J.; Gilmer, J.; Muelly, M.; Goodfellow, I.; Hardt, M.; and Kim, B. 2018. Sanity Checks for Saliency Maps. In *Advances in Neural Information Processing Systems 32*, 11.

Audemard, G.; Bellart, S.; Bounia, L.; Koriche, F.; Lagniez, J.-M.; and Marquis, P. 2022. On the explanatory power of Boolean decision trees. *Data & Knowledge Engineering*, 142: 102088.

Bassan, S.; and Katz, G. 2023. Towards formal XAI: formally approximate minimal explanations of neural networks. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 187–207. Springer.

Bordt, S.; Finck, M.; Raidl, E.; and von Luxburg, U. 2022. Post-Hoc Explanations Fail to Achieve their Purpose in Adversarial Contexts. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, 891–905. ACM.

Chen, C.; Li, O.; Tao, C.; Barnett, A. J.; Su, J.; and Rudin, C. 2019a. *This* Looks like *That*: Deep Learning for Interpretable Image Recognition. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 8930–8941.

Chen, C.; Li, O.; Tao, D.; Barnett, A.; Rudin, C.; and Su, J. K. 2019b. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32.

Chen, Z.; Bei, Y.; and Rudin, C. 2020. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12): 772–782.

Choi, C. L.; Duplessis, A.; and Belongie, S. 2025. Unlearning-based Neural Interpretations. In *The Thirteenth International Conference on Learning Representations*.

Colin, J.; Fel, T.; Cadène, R.; and Serre, T. 2022. What I cannot predict, I do not understand: A human-centered evaluation framework for explainability methods. *Advances in neural information processing systems*, 35: 2832–2845.

Davoodi, O.; Mohammadizadehsamakosh, S.; and Komeili, M. 2023. On the interpretability of part-prototype based classifiers: a human centric analysis. *Scientific Reports*, 13(1): 23088.

De Palma, A.; Durand, S.; Chihani, Z.; Terrier, F.; and Urban, C. 2025. On Using Certified Training towards Empirical Robustness. *Transactions on Machine Learning Research*.

Dijkstra, E. 1976. *A discipline of programming*. Prentice-Hall series in automatic computation. Prentice-Hall. ISBN 0-13-215871-X.

Dombrowski, A.-K.; Alber, M.; Anders, C.; Ackermann, M.; Müller, K.-R.; and Kessel, P. 2019. Explanations Can Be Manipulated and Geometry Is to Blame. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Espinosa Zarlenga, M.; Barbiero, P.; Ciravegna, G.; Marra, G.; Giannini, F.; Diligenti, M.; Shams, Z.; Precioso, F.; Melacci, S.; Weller, A.; et al. 2022. Concept embedding models: Beyond the accuracy-explainability trade-off. *Advances in Neural Information Processing Systems*, 35: 21400–21413.

Fel, T.; Picard, A.; Bethune, L.; Boissin, T.; Vigouroux, D.; Colin, J.; Cadène, R.; and Serre, T. 2023. Craft: Concept recursive activation factorization for explainability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2711–2721.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hedström, A.; Weber, L.; Lapuschkin, S.; and Höhne, M. 2024. A fresh look at sanity checks for saliency maps. In *World Conference on Explainable Artificial Intelligence*, 403–420. Springer.

Helbling, A.; Meral, T. H. S.; Hoover, B.; Yanardag, P.; and Chau, D. H. 2025. ConceptAttention: Diffusion Transformers Learn Highly Interpretable Features. arXiv:2502.04320.

Ignatiev, A.; Narodytska, N.; and Marques-Silva, J. 2019. Abduction-based explanations for machine learning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1511–1519.

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

Katz, G.; Barrett, C.; Dill, D. L.; Julian, K.; and Kochenderfer, M. J. 2017. *Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks*. Springer International Publishing. ISBN 9783319633879.

Kim, B.; Wattenberg, M.; Gilmer, J.; Cai, C.; Wexler, J.; Viegas, F.; et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, 2668–2677. PMLR.

Koh, P. W.; Nguyen, T.; Tang, Y. S.; Mussmann, S.; Pierson, E.; Kim, B.; and Liang, P. 2020. Concept bottleneck models. In *International conference on machine learning*, 5338–5348. PMLR.

Krause, J.; Stark, M.; Deng, J.; and Fei-Fei, L. 2013. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, 554–561.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Lake, B. M.; Ullman, T. D.; Tenenbaum, J. B.; and Gershman, S. J. 2017. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40: e253.

LeCun, Y. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

maintainers, T.; and contributors. 2016. TorchVision: PyTorch's Computer Vision library. https://github.com/pytorch/vision.

Marques-Silva, J.; and Huang, X. 2024. Explainability Is Not a Game. *Communications of the ACM*, 67(7): 66–75.

Marques-Silva, J.; and Ignatiev, A. 2022. Delivering trustworthy AI through formal XAI. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 12342–12350.

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

Molnar, C. 2025. *Interpretable Machine Learning*. 3 edition. ISBN 978-3-911578-03-5.

Montavon, G.; Binder, A.; Lapuschkin, S.; Samek, W.; and Müller, K.-R. 2019. Layer-Wise Relevance Propagation: An Overview. In *Explainable AI*.

Nauta, M.; Trienes, J.; Pathak, S.; Nguyen, E.; Peters, M.; Schmitt, Y.; Schlötterer, J.; Van Keulen, M.; and Seifert, C. 2023. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. *ACM Computing Surveys*, 55(13s): 1–42.

Nauta, M.; van Bree, R.; and Seifert, C. 2021. Neural Prototype Trees for Interpretable Fine-grained Image Recognition. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 14928–14938.

Nilsback, M.-E.; and Zisserman, A. 2008. Automated Flower Classification over a Large Number of Classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*.

Parkhi, O. M.; Vedaldi, A.; Zisserman, A.; and Jawahar, C. V. 2012. Cats and Dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3): 211–252.

Rymarczyk, D.; Struski, Ł.; Górszczak, M.; Lewandowska, K.; Tabor, J.; and Zieliński, B. 2022. Interpretable image classification with differentiable prototypes assignment. In *European Conference on Computer Vision*, 351–368. Springer.

Rymarczyk, D.; Struski, L.; Tabor, J.; and Zieliński, B. 2021. ProtoPShare: Prototypical Parts Sharing for Similarity Discovery in Interpretable Image Classification. 1420–1430.

Sacha, M.; Jura, B.; Rymarczyk, D.; Struski, Ł.; Tabor, J.; and Zieliński, B. 2024. Interpretability Benchmark for Evaluating Spatial Misalignment of Prototypical Parts Explanations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(19): 21563–21573.

Santis, F. D.; Ciravegna, G.; Bich, P.; Giordano, D.; and Cerquitelli, T. 2025. V-CEM: Bridging Performance and Intervenability in Concept-based Models. arXiv:2504.03978.

Shi, W.; Shih, A.; Darwiche, A.; and Choi, A. 2020. On Tractable Representations of Binary Neural Networks. In *International Conference on Principles of Knowledge Representation and Reasoning*.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Bengio, Y.; and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Smilkov, D.; Thorat, N.; Kim, B.; Viégas, F.; and Wattenberg, M. 2017. SmoothGrad: removing noise by adding noise. arXiv:1706.03825.

Soria, J.; Chihani, Z.; Girard-Satabin, J.; Grastien, A.; Xu-Darme, R.; and Cancila, D. 2025. Towards Abductive Latent Explanations. In press.

Su, J.-C.; and Maji, S. 2021. The Semi-Supervised iNaturalist Challenge at the FGVC8 Workshop. arXiv:2106.01364.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 3319–3328. PMLR.

Van Looveren, A.; and Klaise, J. 2021. Interpretable counterfactual explanations guided by prototypes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 650–665. Springer.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.

Willard, F.; Moffett, L.; Mokel, E.; Donnelly, J.; Guo, S.; Yang, J.; Kim, G.; Barnett, A. J.; and Rudin, C. 2024. This Looks Better than That: Better Interpretable Models with ProtoPNeXt. *CoRR*, abs/2406.14675.

Wolf, L.; Galanti, T.; and Hazan, T. 2019. A formal approach to explainability. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 255–261.

Wu, M.; Wu, H.; and Barrett, C. 2023. VeriX: towards verified explainability of deep neural networks. *Advances in neural information processing systems*, 36: 22247–22268.

Wu, M.; et al. 2024. Better Verified Explanations with Applications to Incorrectness and Out-of-Distribution Detection. *arXiv:2409.03060*.

Xu-Darme, R.; Varasse, A.; Grastien, A.; Girard, J.; and Chihani, Z. 2024. CaBRNet, an open-source library for developing and evaluating Case-Based Reasoning Models. arXiv:2409.16693.

Yu, J.; Ignatiev, A.; Stuckey, P. J.; Narodytska, N.; and Marques-Silva, J. 2023. Eliminating the Impossible, Whatever Remains Must Be True: On Extracting and Applying Background Knowledge in the Context of Formal Explanations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4): 4123–4131.

Zagoruyko, S.; and Komodakis, N. 2016. Wide Residual Networks. In *BMVC*.