Causal Explanations in Deep Learning Systems

A thesis submitted in partial fulfillment of the requirements for the award of the degree of

Master of Technology

in

Computer Science

submitted by

Dhruv Vansraj Rathore

(Roll No. CS2306)

Supervisor:

Prof. Utpal Garain

Computer Vision & Pattern Recognition Unit,

Indian Statistical Institute, Kolkata

Indian Statistical Institute

Kolkata-700108, India

June, 2025

Contents

Declaration of Authorship

I, **Dhruv Vansraj Rathore**, declare that this Master's thesis titled, **"Causal Explanations in Deep Learning Systems"** is a result of my own research carried out under the guidance of **Prof. Utpal Garain** and to the best of my knowledge contains no materials previously published or written by any other person, or substantial proportions of material which has formed the basis of award of any other degree or diploma at ISI Kolkata or any other educational institution, except where due acknowledgement is made in this thesis. I authorise ISI Kolkata to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Date: 11 June, 2025

Dhruv Vansraj Rathore
Roll No. CS2306
Indian Statistical Institute, Kolkata

Certificate of the Supervisor

This is to certify that the dissertation titled **"Causal Explanations in Deep Learning Systems"** submitted by **Dhruv Vansraj Rathore** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bonafide record of work carried out by him under our supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in our opinion, has reached the standard needed for submission.

Utpal Garain 10/06/2025

Dr. Utpal Garain
Professor, Computer Vision & Pattern Recognition Unit,
Centre for Artificial Intelligence and Machine Learning
Indian Statistical Institute, Kolkata

Acknowledgements

I am truly grateful to **Prof. Utpal Garain** for giving me the opportunity to work under his supervision during my master's program. I feel fortunate to have learned from his experience and guidance. He not only supported me throughout the journey but also gave me the freedom to explore and learn independently during my dissertation. His thoughtful feedback and encouraging nature played a key role in helping me grow academically and confidently carry out my research.

I would like to sincerely thank **Mr. Saptarshi Saha** for his invaluable support and guidance throughout the course of my dissertation. I genuinely believe that this work would not have been possible without his help. He patiently assisted me in understanding the challenging concepts of causality and guided me through many aspects of implementation and coding. What I deeply appreciate is that, while offering constant support, he also gave me the freedom to learn and explore on my own throughout the course of this work. Even during the stressful period of placements, he selflessly took the time to support me and share my workload. I am truly grateful for his generosity, kindness and constant encouragement, which played a significant role in helping me complete this dissertation.

Finally, I would like to express my heartfelt thanks to my family, friends and batchmates for their unwavering support and encouragement throughout this journey. Their patience, understanding and constant motivation have been a source of strength for me. This achievement would not have been possible without their presence and belief in me at every step.

Abstract

Deep learning models often deliver high predictive accuracy; however, their lack of interpretability can hinder their adoption in critical fields such as healthcare and finance. This thesis explores the concept of Intrinsic Causal Contribution (ICC), a novel method for explaining neural network predictions by quantifying each input feature's intrinsic causal influence on the output, independent of correlated effects. ICC models the network as a Structural Causal Model and employs Causal Normalizing Flows to handle complex dependencies, with efficient estimation via the Jansen Estimator. Analysis on both synthetic and real data sets provides evidence that ICC produces faithful, interpretable attributions, often outperforming traditional approaches like SHAP and LIME. By revealing truly influential features, ICC supports transparent and responsible AI, especially in sensitive settings such as medical diagnosis.

Publications

1. Saptarshi Saha, Dhruv Vansraj Rathore, Soumadeep Saha, Utpal Garain, David Doermann, *"On Measuring Intrinsic Causal Attributions in Deep Neural Networks"*
(Accepted at Causal Learning and Reasoning (CLeaR) 2025)
arXiv:2505.09660

List of Abbreviations

Abbreviation	Full Form
ACE	Average Causal Effect
ADCE	Average Direct Causal Effect
AICE	Average Indirect Causal Effect
CNF	Causal Normalizing Flow
DAG	Directed Acyclic Graph
GAM	Global Attribution Mapping
ICC	Intrinsic Causal Contribution
IG	Integrated Gradients
I$\times$G	Gradient $\times$ Input
LIME	Local Interpretable Model-agnostic Explanations
MAF	Masked Autoregressive Flow
MMD	Maximum Mean Discrepancy
NN	Neural Network
PGI	Prediction Gap on Important Features
PGU	Prediction Gap on Unimportant Features
PFI	Permutation Feature Importance
SCM	Structural Causal Model
SG	SmoothGrad
SHAP	Shapley Additive exPlanations
SP-LIME	Submodular Pick LIME
TMI	Triangular Monotonic Increasing

Chapter 1

Introduction

1.1 Prior Work and Motivation

In recent years, the integration of causal principles into neural network (NN) models has gained significant attention, driven by the need for interpretability and robustness in machine learning systems. Early work by Chattopadhyay et al. [1] laid the foundation for introducing causal attributions in NN models. They represented neural networks as structural causal models (SCMs), enabling the estimation of the average causal impact of input features on the predictions made by the model. This approach provided a causal perspective on how inputs influence outputs, going beyond traditional correlation-based explanations for model interpretability. Expanding on this idea, Reddy et al. [2] explored how NNs can capture indirect causal effects by adding feedforward links between input neurons. This allowed the model to learn and explain not just direct, but also indirect causal relationships between inputs and outputs.

Building on these developments, Janzing et al.[3] introduced the concept of Intrinsic Causal Contribution (ICC), which aims to measure how much a specific input feature intrinsically contributes to the output of a model - removing the influence that might be inherited from other upstream causes. Unlike standard causal effect measures, ICC captures how much a feature contributes to an outcome over and above what it receives from other causally related inputs. To illustrate the importance of ICC in NNs, consider a scenario where we aim to predict an athlete's performance time (P) based on three factors: training intensity (T), genetic predisposition (G) and recovery quality (R). In real-world settings, G has a causal influence on both T and R, and T further affects R. All three - G, T, and R - contribute to determining the final performance outcome P. Despite these causal dependencies, NNs typically treat inputs independently and do not explicitly represent the structural relationships among G, T, and R. Suppose athletes with less favorable genetic traits tend to undergo less intensive training. In such cases, the model might incorrectly associate suboptimal performance solely with lower training intensity, overlooking the genetic constraints that influence both training and recovery. ICC allows us to isolate and quantify the component of T's impact on P that is directly

due to T, disentangled from the effects mediated by G. Such analysis holds practical value in sports science. It enables coaches and practitioners to identify training programs that yield genuine benefits independent of genetic background, thereby promoting fairer and more effective athletic development strategies.

1.2 Contributions of the thesis

Outlined below are the principal contributions of this thesis:

- We explain and elaborate on the concept of *Intrinsic Causal Contribution (ICC)* as defined by Janzing et al. [3], and provide a clear interpretation of its importance in the context of NN explanations.
- We develop an estimation method for computing ICC within NN models, grounded in structural causal modeling principles.
- We validate the reliability and interpretability of ICC through experiments on both synthetic and real-world datasets, showing that it can faithfully capture the intrinsic influence of input features.

1.3 Organisation of thesis

This thesis is organized into five chapters. An overview of each chapter is presented below:

- **Chapter 1:** The chapter presents a summary of the research problem, explains the rationale for conducting the study, and emphasizes the main goals and significant contributions of the work.
- **Chapter 2:** The chapter covers the necessary theoretical background on SCMs and provides a detailed discussion on the definition of ICC.
- **Chapter 3:** The chapter details the proposed methodology, including the computation of ICC and the assumptions and techniques used throughout the study.
- **Chapter 4:** The chapter presents the experimental setup, datasets and implementation details. It also discusses the results, evaluates the performance of the ICC and compares it with some baselines.
- **Chapter 5:** The chapter outlines the main results and highlights the significant contributions made throughout the thesis. It also gives directions for future work.

Chapter 2

Theoretical Foundations

2.1 Structural Causal Models (SCMs)

SCMs, as formalized by Pearl [4], offer a principled framework to represent causal relationships among random variables within a system. In an SCM, the randomness arises solely from unobserved external factors, while observable variables are deterministically generated through causal functions.

Definition 2.1.1 (Structural Causal Model)**.** *A Structural Causal Model is defined as a quadruple $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F}, P_\varepsilon)$, where:*

- $\mathcal{V} = \{X_1, X_2, \ldots, X_n\}$ *is a finite set of* ***endogenous (observable)*** *random variables.*
- $\mathcal{E} = \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$ *is a finite set of* ***exogenous (latent/noise)*** *variables that account for all randomness in the model.*
- $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ *is a set of deterministic functions, where each $X_i \in \mathcal{V}$ is assigned as:*

  $$X_i := f_i(Pa(X_i), \varepsilon_i)$$

  for some subset of variables $Pa(X_i) \subseteq \mathcal{V} \setminus \{X_i\}$.
- P_ε *is a joint probability distribution over the exogenous variables $\mathcal{E}$.*

The functions in $\mathcal{F}$ encode the causal mechanisms of the system, indicating how each variable is generated from its parents and associated noise. Note that we assume acyclicity in the dependency structure, excluding feedback loops from the model.

An SCM $\mathcal{M}$ naturally induces a Directed Acyclic Graph (DAG) $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where:

- Each node in $\mathcal{G}$ corresponds to a variable in $\mathcal{V}$.
- A directed edge $X_j \to X_i$ exists in $\mathcal{A}$ if $X_j \in \text{Pa}(X_i)$, i.e., X_j is a parent of X_i in the structural equation $X_i := f_i(\cdot)$.

This graph $\mathcal{G}$ represents a causal Bayesian network, where the structure encodes the direct causal relationships and the joint distribution over variables can be factorized accordingly based on the graph.

Example An example of an SCM and its associated causal DAG is given below.

$$
\begin{aligned}
X_1 &= \varepsilon_1 \\
X_2 &= f_2(X_1) + \varepsilon_2 \\
X_3 &= f_3(X_1, X_2) + \varepsilon_3 \\
X_4 &= f_4(X_2, X_3) + \varepsilon_4
\end{aligned}
$$

where:

- $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4 \sim \mathcal{N}(0, \sigma^2)$ are exogenous noise variables.
- f_2, f_3, f_4 are deterministic functions, which can be either linear or nonlinear.

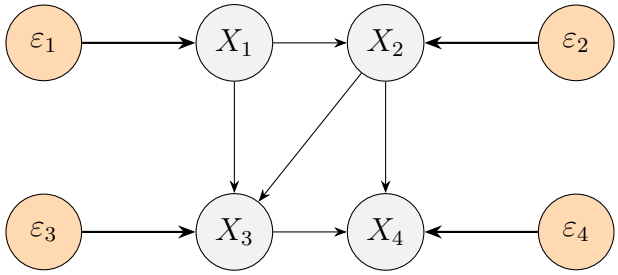


Figure 2.1: Causal DAG showing observed variables X_1-X_4 with their respective exogenous noise terms ε_1 (left of X_1), ε_2 (right of X_2), ε_3 (left of X_3), and ε_4 (right of X_4).

2.2 Neural Networks as SCMs

A neural network trained to predict an output variable Y from input features X can be interpreted as an SCM. The network $\mathcal{N}$, typically optimized by minimizing empirical loss, forms a directed acyclic graph (DAG) where each edge represents information flow between neurons across layers. The prediction $\hat{Y} = \mathcal{N}(X)$ is thus the result of a composition of functional transformations from input to output.

Following the approach of Chattopadhyay et al. [1], we abstract away the hidden layers and focus on the causal relationship between input features and the final output. This allows us to marginalize internal nodes and model the network as a reduced SCM capturing only input-output dependencies.

To quantify ICC [3], we recursively substitute structural equations to express each X_j purely in terms of exogenous noise variables $\mathcal{E}$:

$$X_j = f_j(\text{Pa}(X_j), \varepsilon_j) = F_j(\varepsilon_1, \ldots, \varepsilon_n), \quad \forall\, j \in \{1, \ldots, n\}.$$

2.3 Causal Effects and the *do*-Operator

2.3.1 The *do*-Operator and Interventions

The *do*-operator, introduced by Pearl [4], formalizes interventions in a causal system. An intervention $do(X = x)$ denotes an external manipulation that sets the variable X to a specific value x, breaking any incoming causal influences on X in the original causal graph. The post-intervention distribution $P(Y \mid do(X = x))$ quantifies the effect of setting X to x on another variable Y.

Definition 2.3.1 (Interventional Distribution)**.** *Given a causal model with joint distribution $P(\mathbf{X})$, the interventional distribution under $do(X_i = x_i)$ is defined as:*

$$P(\mathbf{X}_{\setminus i} \mid do(X_i = x_i)) = \prod_{j \neq i} P(X_j \mid pa(X_j))\Big|_{X_i = x_i},$$

where $pa(X_j)$ denotes the parents of X_j in the causal graph.

2.3.2 Average Causal Effect (ACE)

Definition 2.3.2 (Average Causal Effect)**.** *The* Average Causal Effect (ACE) *of a variable X_i on Y is defined as the expected difference in outcomes when intervening to set X_i to two values x and x^*(baseline):*

$$ACE_Y^{X_i} := \mathbb{E}[Y \mid do(X_i = x)] - \mathbb{E}[Y \mid do(X_i = x^*)].$$

ACE captures the total causal effect of X_i on Y, encompassing both direct and indirect pathways in the causal graph.

2.3.3 Direct and Indirect Causal Effects

Let $Z = \text{ch}(X_i)$ be the set of children of X_i excluding the outcome Y. The total causal effect can be decomposed into:

Definition 2.3.3 (Average Direct Causal Effect (ADCE))**.** *The* Average Direct Causal Effect *of X_i on Y, while holding the descendants Z fixed to their baseline values under $X_i = x^*$, is*

given by:

$$ADCE_Y^{X_i} := \mathbb{E}[Y \mid do(X_i = x, Z = Z_{x^*})] - \mathbb{E}[Y \mid do(X_i = x^*, Z = Z_{x^*})],$$

where Z_{x^*} denotes the values of Z induced under the baseline intervention $do(X_i = x^*)$.

Definition 2.3.4 (Average Indirect Causal Effect (AICE))**.** *The* Average Indirect Causal Effect *of X_i on Y through its descendants Z is defined as:*

$$AICE_Y^{X_i} := \mathbb{E}[Y \mid do(X_i = x^*, Z = Z_x)] - \mathbb{E}[Y \mid do(X_i = x^*, Z = Z_{x^*})],$$

where Z_x are the values of Z induced under the intervention $do(X_i = x)$.

By construction, the total causal effect satisfies:

$$ACE_Y^{X_i} = ADCE_Y^{X_i} + AICE_Y^{X_i}.$$

These definitions provide a principled way to decompose the total effect into components that flow directly from X_i to Y and those that are mediated through intermediate variables Z.

For a more detailed treatment of causal effect decompositions - such as average, direct and indirect effect - in the context of neural networks, we refer the reader to the works of Chattopadhyay et al. [1] and Reddy et al. [2], which adopt a principled causal perspective to interpret and explain neural network predictions.

2.4 Defining ICC

We begin by defining the *Intrinsic Causal Contribution* (ICC) [3] of an input variable to the output of a trained neural network model $\mathcal{N}$.

Definition 2.4.1 (Intrinsic Causal Contribution in Neural Networks)**.** *For any feature index $j \in [n]$ and adjustment set $I \subseteq [n] \setminus \{j\}$, the* Intrinsic Causal Contribution *of X_j to $\hat{Y}$ given I is defined as*

$$\mathrm{ICC}_\psi(X_j \to \hat{Y} \mid I) := \psi(\hat{Y} \mid \varepsilon_I) - \psi(\hat{Y} \mid \varepsilon_{I \cup \{j\}}), \tag{2.1}$$

where ψ is a conditional uncertainty measure that satisfies at least one of the following properties:

- ***Monotonicity:*** *$\psi(\hat{Y} \mid \varepsilon_I) \geq \psi(\hat{Y} \mid \varepsilon_{I \cup \{j\}})$, or the reverse inequality;*
- ***Calibration:*** *$\psi(\hat{Y} \mid \varepsilon) = 0$ or $\psi(\hat{Y} \mid \varepsilon_\varnothing) = \psi(\hat{Y}) = 0$.*

Here, ε_I denotes the collection of noise variables $\{\varepsilon_i : i \in I\}$ and $\varnothing$ is the empty set.

For this thesis choose the form of ψ as follows:

$$\psi(\hat{Y} \mid \varepsilon_I) := \frac{\mathbb{V}_{\varepsilon_I}\big(\mathbb{E}[\hat{Y} \mid \varepsilon_I]\big)}{\mathbb{V}[\hat{Y}]} \tag{2.2}$$

We use the above choice of ψ as a measure of conditional uncertainty for the ICC because it directly quantifies the fraction of the total variance in the prediction $\hat{Y}$ that can be attributed to the subset ε_I. Variance is a natural and interpretable measure of uncertainty, robust to estimation errors and expressed in the original units of $\hat{Y}$. Importantly, ψ satisfies the monotonicity property

$$I \subseteq J \implies \psi(\hat{Y} \mid \varepsilon_I) \leq \psi(\hat{Y} \mid \varepsilon_J),$$

which follows directly from the law of total variance.

The intrinsic causal contribution $\mathrm{ICC}_\psi(X_j \to \hat{Y} \mid I)$ depends on the choice of the adjustment set I, which introduces arbitrariness in attribution. To address this, Janzing et al. [3] propose aggregation strategies for ICC based on Shapley values and topological orderings.

2.4.1 Shapley-Based Aggregation of ICC

Formally, the Shapley-aggregated intrinsic causal contribution is given by

$$\mathrm{ICC}_\psi^{\mathrm{Sh}}(X_j \to \hat{Y}) := \sum_{I \subseteq [n] \setminus \{j\}} \frac{1}{n\binom{n-1}{|I|}} \mathrm{ICC}_\psi(X_j \to \hat{Y} \mid I). \tag{2.3}$$

An equivalent and computationally practical formulation expresses this as an average over all permutations π of the input indices:

$$\mathrm{ICC}_\psi^{\mathrm{Sh}}(X_j \to \hat{Y}) := \frac{1}{n!} \sum_{\pi \in S_n} \mathrm{ICC}_\psi(X_j \to \hat{Y} \mid I_\pi^j), \tag{2.4}$$

where $I_\pi^j := \{k \in [n] \setminus \{j\} : \pi(k) < \pi(j)\}$ is the set of features that precede j in the permutation π.

2.4.2 Topological Aggregation of ICC

While Shapley-based ICC provides a principled aggregation, it can be computationally expensive due to summation over all $n!$ permutations. An efficient alternative is to restrict the averaging to causal (topological) orderings of the underlying DAG $\mathcal{G}$.

A topological ordering π is a permutation in which each node precedes all of its descendants. Let $\mathcal{C}(\mathcal{G}) \subseteq S_n$ denote the set of all such valid causal orderings. Then, we define the

topologically averaged ICC as:

$$\mathrm{ICC}_{\psi}^{\mathrm{To}}(X_j \to \hat{Y}) := \frac{1}{|\mathcal{C}(\mathcal{G})|} \sum_{\pi \in \mathcal{C}(\mathcal{G})} \mathrm{ICC}_{\psi}(X_j \to \hat{Y} \mid I_{\pi}^{j}), \tag{2.5}$$

where I_{π}^{j} is defined as in (2.4). This method averages ICC over all valid topological orderings and preserves the causal semantics of the DAG.

While standard causal effects such as ACE, ADCE and AICE are defined through *do*-interventions on endogenous variables, they aim to estimate the total or decomposed influence of an input on the output by breaking existing causal links in the data-generating process. In contrast, ICC is defined through conditioning on exogenous noise variables ε_j, which are not intervened upon but treated as independent sources of variation in an SCM. ICC quantifies the portion of uncertainty in the prediction $\hat{Y}$ that is intrinsically attributable to each input feature via its generating noise, without resorting to the *do*-operator. This distinction makes ICC fundamentally different from conventional interventional approaches: ICC leverages structural invariance and the causal generative process rather than enforcing hypothetical interventions.

Chapter 3

Methodology

This chapter outlines the methodology employed to quantify and compute the ICC of input features to the output of a pre-trained neural network. We leverage a post-hoc framework based on Causal Normalizing Flows (CNFs) [5] to model the causal relationships among input features and extend this framework to incorporate the neural network's predictions. The methodology is detailed across several sections, covering the theoretical foundation, implementation algorithms and evaluation metrics.

3.1 Introduction to the Post-Hoc Framework

The post-hoc framework proposed in this thesis aims to quantify the ICC of input features to the output of a pretrained NN. This approach treats the NN as a deterministic function within a causal framework and utilizes CNFs to model the SCM of the input features. By doing so, we enable the estimation of ICC in a manner that captures the causal influence of each feature on the network's predictions, providing a robust tool for global interpretability. In the next section we first discuss about Normalising Flows.

3.2 Normalizing Flows

Normalizing Flows [6] transform a simple base distribution $P_{\boldsymbol{\varepsilon}}$ (typically a standard Gaussian) into a complex target distribution $P_{\mathbf{X}}$ through an invertible and differentiable mapping $\mathbf{F}_\theta : \mathbb{R}^n \to \mathbb{R}^n$, parameterized by θ. Given $\boldsymbol{\varepsilon} \sim P_{\boldsymbol{\varepsilon}}$, the observed variable $\mathbf{X}$ is defined as:

$$\mathbf{X} = \mathbf{F}_\theta(\boldsymbol{\varepsilon}) \quad \text{and} \quad \boldsymbol{\varepsilon} = \mathbf{F}_\theta^{-1}(\mathbf{X}).$$

Using the change of variables formula, the density $P_{\mathbf{X}}(\mathbf{x})$ is given by:

$$P_{\mathbf{X}}(\mathbf{x}) = P_{\boldsymbol{\varepsilon}}(\mathbf{F}_\theta^{-1}(\mathbf{x})) \left| \det \left(\frac{\partial \mathbf{F}_\theta^{-1}}{\partial \mathbf{x}} \right) \right|,$$

or, equivalently, using the forward map:

$$P_{\mathbf{X}}(\mathbf{x}) = P_{\boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}) \left| \det \left(\frac{\partial \mathbf{F}_\theta}{\partial \boldsymbol{\varepsilon}} \right) \right|^{-1}, \quad \text{where } \boldsymbol{\varepsilon} = \mathbf{F}_\theta^{-1}(\mathbf{x}).$$

In practice, $\mathbf{F}_\theta$ is often constructed as a composition of K simpler invertible transformations:

$$\mathbf{F}_\theta = \mathbf{F}_{\theta_K} \circ \cdots \circ \mathbf{F}_{\theta_1},$$

yielding the log-density:

$$\log P_{\mathbf{X}}(\mathbf{x}) = \log P_{\boldsymbol{\varepsilon}}(\boldsymbol{\varepsilon}) - \sum_{k=1}^{K} \log \left| \det \left(\frac{\partial \mathbf{F}_{\theta_k}}{\partial \boldsymbol{\varepsilon}_{k-1}} \right) \right|,$$

where $\boldsymbol{\varepsilon}_0 = \boldsymbol{\varepsilon}$, $\boldsymbol{\varepsilon}_k = \mathbf{F}_{\theta_k}(\boldsymbol{\varepsilon}_{k-1})$, and $\mathbf{x} = \boldsymbol{\varepsilon}_K$.

To ensure tractable computation of the Jacobian determinant, each component $\mathbf{F}_{\theta_k}$ is typically designed to be either triangular (e.g., autoregressive) or coupling-based. In the context of causal modeling, this triangular structure is further constrained to be monotonic to respect causal ordering - giving rise to the class of Causal Normalizing Flows (CNFs), which align naturally with SCMs.

3.3 Modeling SCMs with CNFs

CNFs provide a powerful and flexible framework for modeling SCMs in a way that respects both the causal structure and the distributional complexity of data. In an SCM, each variable X_j is produced based on a collection of latent noise variables $\boldsymbol{\varepsilon} = \{\varepsilon_1, \ldots, \varepsilon_n\}$ through deterministic structural equations governed by a directed acyclic graph (DAG) encoding causal relationships.

3.3.1 Mathematical Foundation of CNFs

Formally, a CNF is a parameterized, invertible transformation $\mathbf{F}_\theta : \mathbb{R}^n \to \mathbb{R}^n$, where θ denotes the learned parameters. This map transforms independent latent noise variables $\boldsymbol{\varepsilon} \sim P_\varepsilon$ into the observed variables $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$, such that:

$$\mathbf{X} = \mathbf{F}_\theta(\boldsymbol{\varepsilon}) \quad \text{and} \quad \boldsymbol{\varepsilon} = \mathbf{F}_\theta^{-1}(\mathbf{X}).$$

To align with causal semantics, $\mathbf{F}_\theta$ is modeled as a triangular monotonic increasing (TMI) transformation that adheres to a causal ordering. This means each component $F_{\theta,j}$ depends only on the first j components of $\boldsymbol{\varepsilon}$, i.e.,

$$\mathbf{F}_\theta(\boldsymbol{\varepsilon}) = \left[F_{\theta,1}(\varepsilon_1),\ F_{\theta,2}(\varepsilon_1, \varepsilon_2),\ \ldots,\ F_{\theta,n}(\varepsilon_1, \ldots, \varepsilon_n) \right]^\top,$$

with each $F_{\theta,j}$ strictly increasing in ε_j. This triangular structure enforces the causal ordering, ensuring that the generation of each X_j only relies on its causal predecessors and an associated noise variable.

Example. Consider a simple SCM with four variables X_1, X_2, X_3, X_4 defined as:

$$\begin{aligned} X_1 &= \varepsilon_1, \\ X_2 &= f_2(X_1) + \varepsilon_2, \\ X_3 &= f_3(X_1, X_2) + \varepsilon_3, \\ X_4 &= f_4(X_2, X_3) + \varepsilon_4, \end{aligned}$$

where $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4 \sim \mathcal{N}(0, 1)$ are independent noise variables. This system can be represented using a triangular transformation $\mathbf{F}_\theta$ such that:

$$\begin{aligned} X_1 &= F_1(\varepsilon_1), \\ X_2 &= F_2(\varepsilon_1, \varepsilon_2), \\ X_3 &= F_3(\varepsilon_1, \varepsilon_2, \varepsilon_3), \\ X_4 &= F_4(\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4). \end{aligned}$$

This formulation ensures that:

- X_1 depends only on ε_1,
- X_2 depends on $\varepsilon_1, \varepsilon_2$,
- X_3 depends on $\varepsilon_1, \varepsilon_2, \varepsilon_3$,
- X_4 depends on all four noise variables $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4$.

This triangular structure naturally follows the causal ordering $X_1 \prec X_2 \prec X_3 \prec X_4$, and can be effectively captured using a CNF that respects the autoregressive and monotonic constraints.

Such a CNF can be trained to learn $\mathbf{F}_\theta$ so that samples from the latent space $\boldsymbol{\varepsilon}$ can be deterministically transformed into observational data $\mathbf{X}$, while preserving the causal factorization of the joint distribution.

3.4 Augmented SCM for Neural Networks

To integrate the neural network's output into the causal framework, we augment the SCM of the input features. Let $\mathcal{N}$ denote the pre-trained neural network, and $\hat{Y} = \mathcal{N}(\mathbf{X})$ its output. We define an augmented causal graph $\overline{\mathcal{G}} = (\mathcal{V}, \overline{\mathcal{E}})$, where $\mathcal{V} = \mathbf{X} \cup \{\hat{Y}\}$ and $\overline{\mathcal{E}} = \mathcal{E} \cup \bigcup_{j=1}^{p} \{(X_j, \hat{Y})\}$.

Here, $\mathcal{E}$ represents the edges in the original causal graph $\mathcal{G}$ of the input features and the additional edges from each X_j to $\hat{Y}$ reflect the deterministic dependence of the output on all input features via the neural network.

In terms of latent variables, each input feature X_j is a function of a noise variable ε_j (i.e., $X_j = f_j(\text{pa}(X_j), \varepsilon_j)$), where $\text{pa}(X_j)$ denotes the parents of X_j in $\mathcal{G}$. By augmenting the SCM with $\hat{Y} = \mathcal{N}(\mathbf{X})$, we are able to trace how variations in the latent noise variables $\boldsymbol{\varepsilon}$ influence the model's prediction $\hat{Y}$, enabling causal interpretability of neural network behavior.

3.5 Estimation of ψ Using the Jansen Estimator

To compute the ICC, we first need to estimate the uncertainty measure

$$\psi(\hat{Y} \mid \boldsymbol{\varepsilon}_I) = \frac{\mathbb{V}_{\boldsymbol{\varepsilon}_I}\left(\mathbb{E}[\hat{Y} \mid \boldsymbol{\varepsilon}_I]\right)}{\mathbb{V}(\hat{Y})},$$

which quantifies the proportion of output variance attributable to the subset of latent noise variables $\boldsymbol{\varepsilon}_I$. Calculating $\mathbb{V}_{\boldsymbol{\varepsilon}_I}\left(\mathbb{E}[\hat{Y} \mid \boldsymbol{\varepsilon}_I]\right)$ requires computing two nested expectations, which can be computationally expensive. For this we employ the Jansen estimator [7] with a Monte Carlo-based strategy known for its efficiency and low computational cost in variance-based sensitivity analysis.

3.5.1 Mathematical Formulation of the Jansen Estimator

The Jansen estimator provides an efficient approximation of

$$\mathbb{V}_{\boldsymbol{\varepsilon}_I}\left(\mathbb{E}[\hat{Y} \mid \boldsymbol{\varepsilon}_I]\right)$$

using two independent sets of samples. For a subset I, it leverages the difference between outputs when components in I are fixed versus varied. Let $\boldsymbol{\varepsilon}_A$ and $\boldsymbol{\varepsilon}_B$ be two independent samples drawn from the prior P_θ. Construct a hybrid sample $\boldsymbol{\varepsilon}_C$ by combining the components of $\boldsymbol{\varepsilon}_B$ corresponding to I with the components of $\boldsymbol{\varepsilon}_A$ corresponding to the complement of I:

$$\boldsymbol{\varepsilon}_C = (\boldsymbol{\varepsilon}_{A_{-I}}, \boldsymbol{\varepsilon}_{B_I}).$$

The conditional expectation is approximated as:

$$\mathbb{E}[\hat{Y} \mid \boldsymbol{\varepsilon}_I = \boldsymbol{\varepsilon}_{B_I}] \approx \hat{Y}(\boldsymbol{\varepsilon}_C),$$

where the latent noise variables in I are fixed to $\boldsymbol{\varepsilon}_{B_I}$, and the remaining components are drawn independently.

The estimator for the conditional variance is:

$$\hat{V}_I = \frac{1}{2N} \sum_{i=1}^{N} \left(\hat{Y}(\boldsymbol{\varepsilon}_B^{(i)}) - \hat{Y}(\boldsymbol{\varepsilon}_C^{(i)}) \right)^2,$$

here N denotes the number of Monte Carlo samples and each $\boldsymbol{\varepsilon}_C^{(i)}$ is constructed from $\boldsymbol{\varepsilon}_A^{(i)}$ and $\boldsymbol{\varepsilon}_B^{(i)}$ as described above.

The total variance is estimated as:

$$\hat{V} = \frac{1}{2N-1} \sum_{i=1}^{N} \left[\left(\hat{Y}(\boldsymbol{\varepsilon}_A^{(i)}) - \bar{Y} \right)^2 + \left(\hat{Y}(\boldsymbol{\varepsilon}_B^{(i)}) - \bar{Y} \right)^2 \right],$$

where the empirical mean $\bar{Y}$ is estimated as:

$$\bar{Y} = \frac{1}{2N} \sum_{i=1}^{N} \left(\hat{Y}(\boldsymbol{\varepsilon}_A^{(i)}) + \hat{Y}(\boldsymbol{\varepsilon}_B^{(i)}) \right).$$

3.5.2 Step-by-Step Algorithm for Estimating ψ

The following procedure outlines the estimation of ψ for a specified subset of input features indexed by I. The algorithm assumes the availability of: batch size B, conditioning context I, a trained CNF model$(\mathbf{F}_\theta, P_\theta)$ and a predictive neural network $\mathcal{N}$.

1. **Sampling:** For each $i = 1, \ldots, B$ draw two independent latent vectors $\boldsymbol{\varepsilon}_A^{(i)}$ and $\boldsymbol{\varepsilon}_B^{(i)}$ from the prior distribution P_θ. Construct a hybrid latent vector $\boldsymbol{\varepsilon}_C^{(i)}$ by combining the non-I components of $\boldsymbol{\varepsilon}_A^{(i)}$ with the I-components of $\boldsymbol{\varepsilon}_B^{(i)}$:

   $$\boldsymbol{\varepsilon}_C^{(i)} = \left(\boldsymbol{\varepsilon}_{A_{-I}}^{(i)}, \boldsymbol{\varepsilon}_{B_I}^{(i)} \right),$$

   where $-I$ denotes the complement of the index set I.

2. **Forward Transformation and Prediction:** Pass each latent vector through the CNF to obtain data samples:

   $$\mathbf{x}_A^{(i)} = \mathbf{F}_\theta(\boldsymbol{\varepsilon}_A^{(i)}), \quad \mathbf{x}_B^{(i)} = \mathbf{F}_\theta(\boldsymbol{\varepsilon}_B^{(i)}), \quad \mathbf{x}_C^{(i)} = \mathbf{F}_\theta(\boldsymbol{\varepsilon}_C^{(i)}).$$

   Evaluate the neural network $\mathcal{N}$ on these samples to obtain the corresponding predictions:

   $$\hat{y}_A^{(i)} = \mathcal{N}(\mathbf{x}_A^{(i)}), \quad \hat{y}_B^{(i)} = \mathcal{N}(\mathbf{x}_B^{(i)}), \quad \hat{y}_C^{(i)} = \mathcal{N}(\mathbf{x}_C^{(i)}).$$

3. **Total Variance Estimation:** Compute the empirical mean of predictions:

   $$\bar{y} = \frac{1}{2B} \sum_{i=1}^{B} \left(\hat{y}_A^{(i)} + \hat{y}_B^{(i)} \right),$$

and estimate the total variance as:

$$\hat{V}_{\text{total}} = \frac{1}{2B-1} \sum_{i=1}^{B} \left[\left(\hat{y}_A^{(i)} - \bar{y} \right)^2 + \left(\hat{y}_B^{(i)} - \bar{y} \right)^2 \right].$$

4. **Conditional Variance Estimation (Jansen's Estimator):** Use Jansen's estimator to estimate the conditional variance due to subset I as:

$$\hat{V}_I = \frac{1}{2B} \sum_{i=1}^{B} \left(\hat{y}_B^{(i)} - \hat{y}_C^{(i)} \right)^2.$$

This measures the expected change in output when the variables in subset I are varied while the complement $-I$ is held fixed.

5. **Estimate of ψ:** The estimated $\hat{\psi}$ is given by:

$$\hat{\psi} = \frac{\hat{V}_I}{\hat{V}_{\text{total}}}.$$

3.6 Causal Normalizing Flow Implementation

3.6.1 Autoregressive Modeling via Masked Autoregressive Flows

Masked Autoregressive Flows (MAF) are a key component in implementing CNFs [8]. MAF layers construct the transformation $\mathbf{F}_\theta$ by modeling each output dimension as a function of preceding dimensions, adhering to the autoregressive property. A neural network predicts the parameters of each transformation based on prior variables, ensuring the TMI property and invertibility. Stacking multiple MAF layers enables the modeling of complex dependencies, making them an effective choice for our CNF framework.

3.6.2 Training Objective using Maximum Mean Discrepancy Loss

The CNF is trained to approximate the data-generating process of $\mathbf{X}$ using Maximum Mean Discrepancy (MMD) loss [9] and [10]. MMD quantifies the difference between the true distribution P and the generated distribution Q within a reproducing kernel Hilbert space (RKHS)

$$\text{MMD}(P, Q) = \|\mathbb{E}_P[\Psi(\mathbf{x})] - \mathbb{E}_Q[\Psi(\mathbf{x})]\|_{\mathcal{H}}^2,$$

where Ψ is the RKHS feature map. Minimizing MMD aligns the CNF-generated samples with the true distribution.

3.6.3 Model Evaluation via Wasserstein Distance

The quality of the trained CNF is evaluated using the 1-Wasserstein distance [11]:

$$W_1(P, Q) = \inf_{\gamma \in \Gamma(P,Q)} \int_{\mathbb{R}^p \times \mathbb{R}^p} \|\mathbf{x} - \mathbf{y}\|_1 \, d\gamma(\mathbf{x}, \mathbf{y}),$$

where $\Gamma(P, Q)$ denotes joint distributions with marginals P and Q. A lower distance indicates better alignment between the generated and true distributions.

3.6.4 Handling Discrete Variables in CNFs

CNFs are inherently designed for continuous data, which poses a challenge when modeling systems that include discrete variables. In order to deal with it we adopt the method proposed by Xi et.al. [12], which transforms discrete variables into continuous representations by injecting noise.

For a discrete variable X_j, we assume that it corresponds to the integer part of a continuous latent variable $\tilde{X}_j$ perturbed by additive noise:

$$X_j = \lfloor \tilde{X}_j + \epsilon_j \rfloor, \quad \text{where} \quad \epsilon_j \sim \mathcal{U}[0, 1].$$

Here, $\tilde{X}_j$ is a continuous variable generated from an SCM that adheres to our modeling assumptions and ϵ_j is independently drawn uniform noise in $[0, 1]$.

This approach ensures that the original discrete distribution is recoverable from the continuous model, thereby allowing CNFs to treat all variables as continuous during training and inference. Crucially, it maintains the integrity of the underlying distribution while enabling the application of continuous-flow-based methods to hybrid discrete-continuous data.

Chapter 4

Experiments and Results

4.1 Datasets

We utilized three datasets to perform experiments. For each dataset, we divided the data into train (75%), validation (10%) and test (15%) subsets.

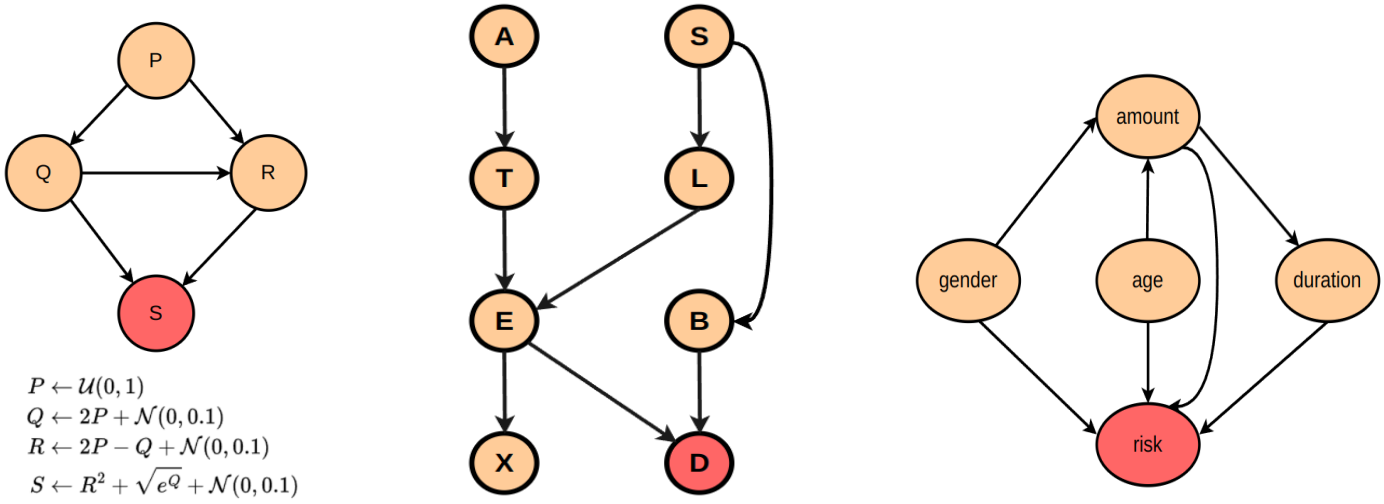
Synthetic Dataset: We generate a similar synthetic dataset as used in [2] but with different causal mechanisms as shown in Figure 4.1(a). It comprises a total of 10,000 samples. We treat P, Q and R as input features and S as the output variable, with the task formulated as a regression problem.

ASIA Dataset: This dataset is a classical probabilistic graphical model used in causal reasoning studies [13]. It contains 10,000 samples generated from the ASIA Bayesian Network [14]. The dataset consists of the following binary variables:

- VisitAsia (A) : recent travel to Asia
- Tuberculosis (T) : presence of tuberculosis
- Smoking (S): smoking status
- LungCancer (L): presence of lung cancer
- Bronchitis (B): presence of bronchitis
- Either (E): either lung cancer or tuberculosis
- XRay (X) : abnormal X-ray result
- Dyspnea (D) : presence of shortness of breath

The task is formulated as a binary classification problem where the target variable is Dyspnea (D).

German Credit Dataset: This dataset contains 1,000 samples, each representing a loan applicant described by a set of socio-economic and financial features[15]. We consider a subset of input features and the causal graph as used in previous studies [16] and [17]. The selected features include Age, Credit amount, Duration, Gender and the target variable Risk (indicating good or bad credit risk). The task is a binary classification problem.



(a) Synthetic (b) ASIA (c) German Credit

Figure 4.1: Causal graphs.

4.2 Experimental Setup

We train both a standard NN and an SCM on each dataset. The SCM is instantiated using CNFs, which provide a flexible and expressive framework for modeling the causal generative process of the input features.

All models are implemented in `PyTorch Lightning` [18] to facilitate modular, scalable, and reproducible training. For the CNF-based SCM, we use the `Zuko` library [19], leveraging the MAF as the core flow architecture. The autoregressive nature of MAF aligns naturally with the triangular structure of SCMs, capturing the causal dependencies among variables through a predefined ordering.

The NN and CNF models are trained with a batch size of 64 for up to 500 epochs, employing early stopping based on validation performance to avoid overfitting. The Adam optimizer is used for optimization with a learning rate of 3×10^{-4}. The CNF is trained using the MMD loss function.

To ensure stability and reproducibility, all experiments are conducted using three different random seeds. Training is performed on a single Nvidia RTX A5000 GPU, utilizing `PyTorch Lightning`'s native support for GPU acceleration.

The architectural configuration for both the CNF and the NN consists of two hidden layers with 256 units each for the flow's autoregressive transformation and the prediction NN layers. The CNF employs a standard multivariate normal distribution as its base.

4.3 Results from Trained Models

4.3.1 Quantitative Results

Performance of Trained NN and SCM Models

Table 4.1 and 4.2 and presents the main metrics for both the trained NN and CNF-based SCM models. The metrics are averaged across three different seeds.

Table 4.1: Performance of neural network $\mathcal{N}$ across datasets.

Task Type	**Dataset**	**Metric Value**
Regression	Synthetic	RMSE: $0.0974_{\pm 0.0004}$
Classification	ASIA	Accuracy: $0.8082_{\pm 0.0002}$
	German-Credit	Accuracy: $0.7133_{\pm 0.0031}$

Table 4.2: **1-Wasserstein distance** ($\mathcal{W}_1$) between CNF-generated features and test features.

Dataset	**$\mathcal{W}_1$-Distance**
Synthetic	$0.1407_{\pm 0.0064}$
ASIA	$0.5373_{\pm 0.0033}$
German-Credit	$0.3789_{\pm 0.0236}$

Wasserstein Distance Analysis

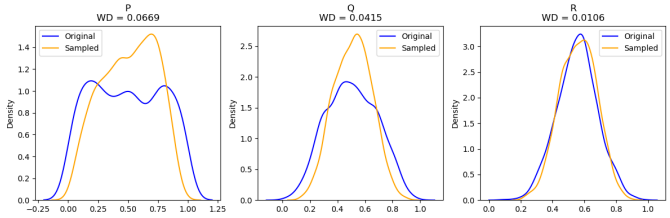
To quantify the closeness between marginal distributions, Table 4.3 reports the 1D Wasserstein distances for each feature. These distances are computed between the test data and samples generated by the SCM model.

4.3.2 Qualitative Results

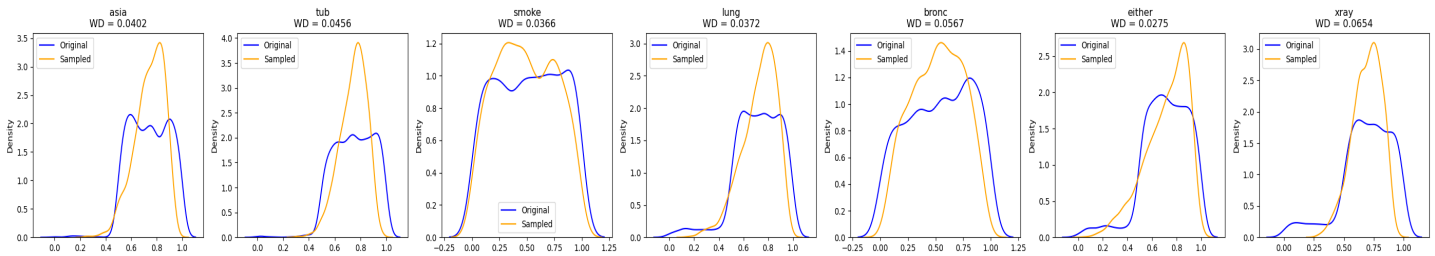
Figure 4.2 shows dimension-wise KDE plots comparing SCM-generated data with the corresponding test data. The KDE curves indicate a close match between the distributions, suggesting that the generative model captures key statistical properties of the test data. To further evaluate sample alignment, Figure 4.3 presents KDE plots over 2D PCA projections of both test and generated samples.

Dataset	Variable	Wasserstein Distance
Syn	P	$0.0704_{\pm 0.0041}$
	Q	$0.0367_{\pm 0.002}$
	R	$0.0093_{\pm 0.0012}$
ASIA	Asia	$0.0408_{\pm 0.0009}$
	Tub	$0.0404_{\pm 0.0031}$
	Smoke	$0.0337_{\pm 0.0014}$
	Lung	$0.0347_{\pm 0.0016}$
	Bronc	$0.0459_{\pm 0.0045}$
	Either	$0.0307_{\pm 0.0019}$
	Xray	$0.0548_{\pm 0.0111}$
German	Gender	$0.0871_{\pm 0.0021}$
	Age	$0.0432_{\pm 0.0034}$
	Amount	$0.0453_{\pm 0.0032}$
	Duration	$0.1369_{\pm 0.02}$

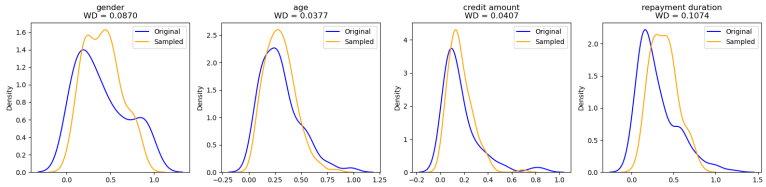
Table 4.3: Dimension-wise Wasserstein distances.



(a) Synthetic dataset



(b) ASIA dataset

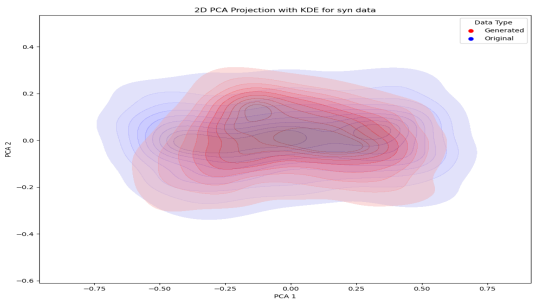


(c) German Credit dataset

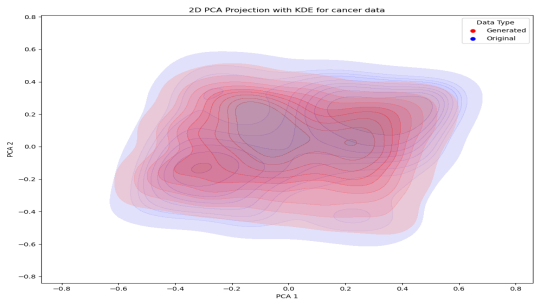
Figure 4.2: Dimensionwise KDE plots comparing real and SCM-generated data.

4.4 Generating Global Attribution Explanations

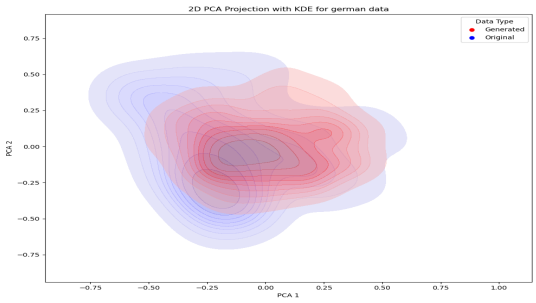
We compare ICC with established global attribution methods, including GAM [20], SP-LIME [21], and Permutation Feature Importance (PFI) [22]. For comprehensive analysis, GAM is ap-



(a) Synthetic dataset



(b) ASIA dataset



(c) German Credit dataset

Figure 4.3: 2D PCA-based KDE plots comparing real and SCM-generated samples.

plied to five local attribution techniques: Integrated Gradients (IG) [23], Gradient $\times$ Inputs (I$\times$G) [24], SmoothGrad (SG) [25], Shapley Values (SHAP) [26], and LIME [21]. This generates global attribution summaries across all test samples. We use the `OpenXAI` codebase [27] to generate all these attribution explanations. The short details of these local attribution techniques are given in Appendix.

Figure 4.4 visualizes global attribution patterns for the three test datasets. In the synthetic dataset (top-left), both ICC methods assign comparatively lower attributions to feature P, which aligns with the ground truth where P has only a limited direct effect on the outcome. In the ASIA dataset (top-right), ICC SHAP and ICC TOP both assigns the highest attribution to the `bronc` node, sharply distinguishing its importance due to its strong downstream influence on the target variable. Other methods such as LIME and SP-LIME also highlight `bronc` as the most influential.

In the German Credit dataset (bottom), ICC TOP assigns dominant importance to `repayment duration`, distinctly separating it from the other features. Traditional attribution methods like SHAP and LIME tend to spread importance across multiple features, leading to less decisive explanations.

4.4.1 Evaluating Attribution Faithfulness

We assess predictive faithfulness using two complementary metrics as previously used in [27]:

- **Prediction Gap on Important features (PGI) [27]**: Measures model sensitivity when perturbing influential features

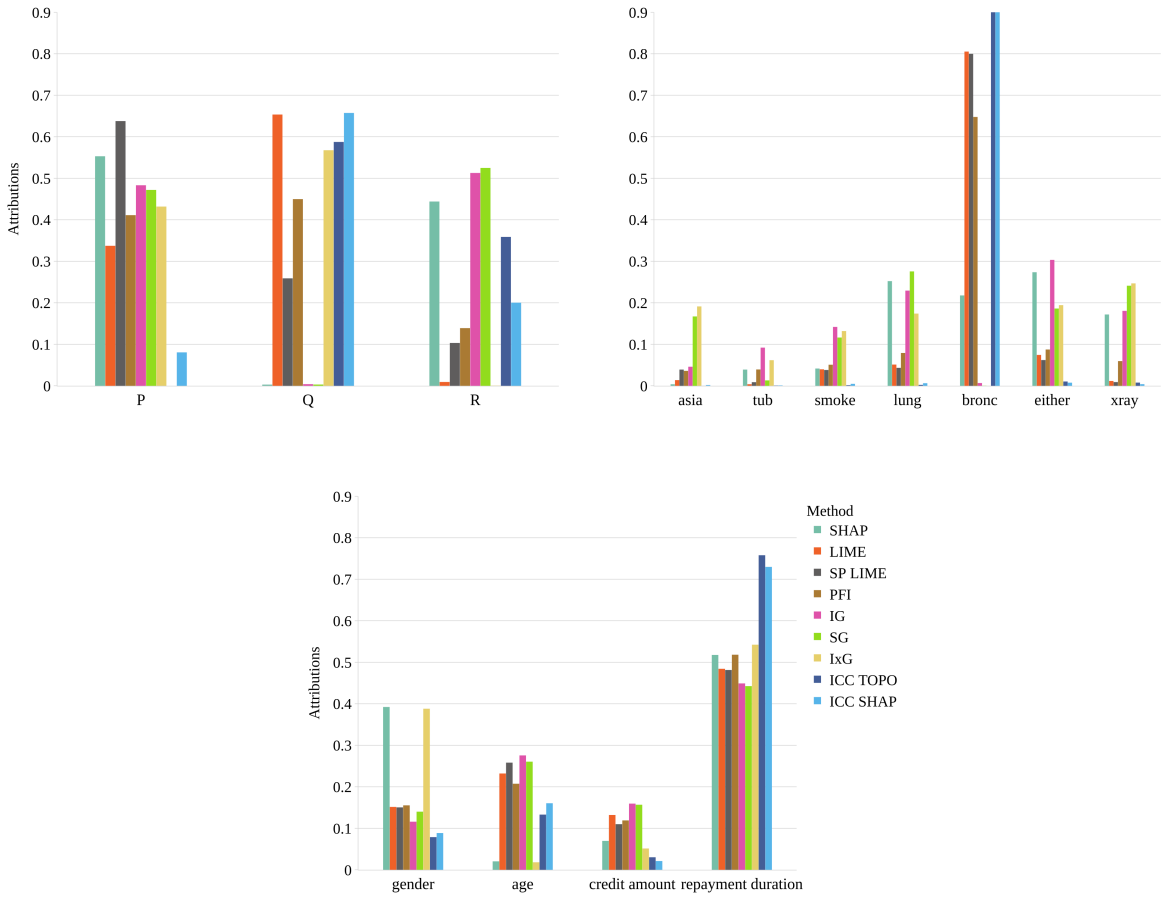


Figure 4.4: Global attribution explanations

- **Prediction Gap on Unimportant features (PGU) [27]**: Measures model robustness when perturbing non-influential features

For an input $\mathbf{x}$ with prediction $\hat{y} = f(\mathbf{x})$ and explanation $e_{\mathbf{x}}$:

$$\text{PGI}(\mathbf{x}, f, e_{\mathbf{x}}, k) = \mathbb{E}_{\mathbf{x}'}\left[|\hat{y} - f(\mathbf{x}')|\right] \quad \text{where} \quad \mathbf{x}' \sim \mathcal{P}_{\text{top-}k}(\mathbf{x}) \tag{4.1}$$

$$\text{PGU}(\mathbf{x}, f, e_{\mathbf{x}}, k) = \mathbb{E}_{\mathbf{x}'}\left[|\hat{y} - f(\mathbf{x}')|\right] \quad \text{where} \quad \mathbf{x}' \sim \mathcal{P}_{\text{non-top-}k}(\mathbf{x}) \tag{4.2}$$

where $\mathcal{P}$ generates perturbed instances by adding Gaussian/Uniform noise to either top-k important features (PGI) or non-top-k features (PGU) while fixing others.

We report two aggregate measures:

1. **AUC**: Area under the curve for $k \in \{1, 2, ..., d\}$
2. **SUM**: $\sum_{k=1}^{d} \text{PGI}(\cdot, k)$ and $\sum_{k=1}^{d} \text{PGU}(\cdot, k)$

Higher PGI and lower PGU values indicate better faithfulness. Tables 4.4, 4.5 and 4.6 report PGI/PGU metrics (mean±standard error) aggregated across test instances for each respective dataset.

Metric	SHAP	LIME	SP-LIME	PFI	IG	SG	I×G	ICC^{To}	ICC^{Sh}
PGI AUC	$0.1080_{\pm 0.0006}$	$0.1457_{\pm 0.0007}$	$0.1175_{\pm 0.0006}$	$0.1455_{\pm 0.0007}$	$0.1081_{\pm 0.0006}$	$0.1081_{\pm 0.0006}$	$0.1456_{\pm 0.0008}$	$\mathbf{0.1914}_{\pm 0.0010}$	$0.1913_{\pm 0.0010}$
PGU AUC	$0.1376_{\pm 0.0008}$	$0.0541_{\pm 0.0006}$	$0.0816_{\pm 0.0004}$	$0.0541_{\pm 0.0006}$	$0.1148_{\pm 0.0007}$	$0.1147_{\pm 0.0007}$	$0.0540_{\pm 0.0006}$	$\mathbf{0.0538}_{\pm 0.0004}$	$\mathbf{0.0538}_{\pm 0.0004}$
PGI SUM	$0.3307_{\pm 0.0013}$	$0.4624_{\pm 0.0022}$	$0.3497_{\pm 0.0015}$	$0.4618_{\pm 0.0022}$	$0.3310_{\pm 0.0016}$	$0.3310_{\pm 0.0016}$	$0.4623_{\pm 0.0022}$	$\mathbf{0.5539}_{\pm 0.0027}$	$0.5537_{\pm 0.0027}$
PGU SUM	$0.3810_{\pm 0.0021}$	$0.1587_{\pm 0.0017}$	$0.2691_{\pm 0.0013}$	$0.1588_{\pm 0.0017}$	$0.2897_{\pm 0.0017}$	$0.2896_{\pm 0.0017}$	$0.1586_{\pm 0.0017}$	$\mathbf{0.1583}_{\pm 0.0013}$	$0.1582_{\pm 0.0013}$

Table 4.4: PGI/PGU AUC and SUM for the Synthetic dataset (scaled by 10^{-1}).

Metric	SHAP	LIME	SP-LIME	PFI	IG	SG	I×G	ICC^{To}	ICC^{Sh}
PGI AUC	$0.1976_{\pm 0.0041}$	$0.2216_{\pm 0.0047}$	$0.1578_{\pm 0.0019}$	$\mathbf{0.2258}_{\pm 0.0048}$	$0.1471_{\pm 0.0025}$	$0.1544_{\pm 0.0024}$	$0.1610_{\pm 0.0022}$	$0.2248_{\pm 0.0047}$	$0.2255_{\pm 0.0048}$
PGU AUC	$0.1278_{\pm 0.0014}$	$0.1211_{\pm 0.0017}$	$0.1653_{\pm 0.0032}$	$0.1162_{\pm 0.0017}$	$0.1893_{\pm 0.0033}$	$0.1976_{\pm 0.0031}$	$0.1992_{\pm 0.0034}$	$\mathbf{0.1138}_{\pm 0.0016}$	$0.1176_{\pm 0.0016}$
PGI SUM	$1.3321_{\pm 0.0272}$	$1.5534_{\pm 0.0320}$	$1.0930_{\pm 0.0137}$	$\mathbf{1.5781}_{\pm 0.0327}$	$1.0291_{\pm 0.0173}$	$1.0706_{\pm 0.0164}$	$1.1163_{\pm 0.0153}$	$1.5725_{\pm 0.0321}$	$1.5763_{\pm 0.0327}$
PGU SUM	$0.8733_{\pm 0.0092}$	$0.8144_{\pm 0.0110}$	$1.1037_{\pm 0.0214}$	$0.7851_{\pm 0.0112}$	$1.2420_{\pm 0.0221}$	$1.2854_{\pm 0.0204}$	$1.3048_{\pm 0.0223}$	$\mathbf{0.7707}_{\pm 0.0108}$	$0.7932_{\pm 0.0108}$

Table 4.5: PGI/PGU AUC and SUM for the ASIA dataset

Metric	SHAP	LIME	SP-LIME	PFI	IG	SG	I×G	ICC^{To}	ICC^{Sh}
PGI AUC	$0.2005_{\pm 0.0015}$	$0.2061_{\pm 0.0017}$	$0.1389_{\pm 0.0014}$	$0.2063_{\pm 0.0016}$	$\mathbf{0.2104}_{\pm 0.0015}$	$\mathbf{0.2104}_{\pm 0.0015}$	$0.2008_{\pm 0.0015}$	$0.2065_{\pm 0.0017}$	$0.2066_{\pm 0.0017}$
PGU AUC	$0.1175_{\pm 0.0012}$	$0.0964_{\pm 0.0007}$	$0.1694_{\pm 0.0012}$	$0.0962_{\pm 0.0007}$	$0.0794_{\pm 0.0011}$	$\mathbf{0.0793}_{\pm 0.0011}$	$0.1176_{\pm 0.0012}$	$0.0964_{\pm 0.0008}$	$0.0964_{\pm 0.0007}$
PGI SUM	$0.8060_{\pm 0.0060}$	$0.8231_{\pm 0.0065}$	$0.5491_{\pm 0.0056}$	$0.8236_{\pm 0.0065}$	$\mathbf{0.8368}_{\pm 0.0062}$	$0.8364_{\pm 0.0062}$	$0.8078_{\pm 0.0061}$	$0.8244_{\pm 0.0066}$	$0.8247_{\pm 0.0065}$
PGU SUM	$0.4300_{\pm 0.0042}$	$0.3669_{\pm 0.0028}$	$0.6166_{\pm 0.0043}$	$0.3662_{\pm 0.0028}$	$0.3158_{\pm 0.0038}$	$\mathbf{0.3155}_{\pm 0.0039}$	$0.4304_{\pm 0.0041}$	$0.3663_{\pm 0.0028}$	$0.3669_{\pm 0.0027}$

Table 4.6: PGI/PGU AUC and SUM for the German Credit dataset

Across all three datasets, the ICC-based methods (ICC^{To} and ICC^{Sh}) perform strongly in identifying important features using PGI and PGU metrics. In the Synthetic dataset (4.4),ICC^{To} achieve the highest PGI and lowest PGU scores, clearly outperforming other methods showing that ICC^{To} assign more focused importance to truly predictive features.

In the ASIA dataset (4.5), ICC^{To} gives the lowest PGU AUC and SUM scores with ICC^{Sh} close as well.Thus they give less importance to uninformative features and produce more reliable explanations. We also see PFI with the highest PGI AUC and SUM scores.

In the German Credit dataset (4.6), the ICC methods are not the top performers. Overalll IG and SG score slightly higher - but ICCs still rank well, placing third (ICC^{Sh}) and fourth (ICC^{To}) in PGI scores. They continue to show low PGU AUC and SUM values, indicating that they avoid attributing importance to irrelevant inputs.

Chapter 5

Conclusion and Future Works

To sum up, this thesis has explored a causal approach to explaining neural network predictions by introducing the concept of Intrinsic Causal Contribution (ICC), situated within the broader field of explainable AI. We modeled neural networks as Structural Causal Models (SCMs) and applied Causal Normalizing Flows (CNFs) to capture complex dependencies among input features. Using the Jansen estimator, we efficiently computed the intrinsic causal effect of individual features on model outputs. We assessed the effectiveness of ICC by conducting thorough experiments on both synthetic and real-world datasets, comparing it with popular global attribution techniques. Our analysis revealed that ICC consistently offers more faithful and interpretable explanations by isolating true causal influence from spurious correlations. Overall this thesis lays the groundwork for future research in causal AI and responsible machine learning.

A promising extension of this work lies in adapting the ICC framework to high-dimensional image data. While ICC has proven effective for tabular datasets by capturing the intrinsic causal influence of input features, its application to pixel-level interactions in vision models presents scalability challenges, especially due to the factorial complexity of Shapley-based aggregation. Drawing inspiration from the Sobol-based sensitivity analysis presented in the paper [28], future work could explore mask-based perturbation strategies over image regions - such as superpixels or latent - space patches - combined with efficient Quasi-Monte Carlo sampling [29] to reduce the estimation complexity of ICC. Evaluating this extension on established computer vision benchmarks such as the Pointing Game [30] or Deletion metrics [31] could highlight the advantages of causal attributions, particularly in domains like medical imaging where interpretability and causal faithfulness are critical.

Bibliography

[1] A. Chattopadhyay, P. Manupriya, A. Sarkar, and V. N. Balasubramanian, "Neural network attributions: A causal perspective," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 981–990, PMLR, 09–15 Jun 2019.

[2] A. G. Reddy, S. Bachu, H. N. Pathak, B. Godfrey, V. N. Balasubramanian, V. Varshaneya, and S. N. Kar, "Towards learning and explaining indirect causal effects in neural networks," in *AAAI Conference on Artificial Intelligence*, 2023.

[3] D. Janzing, P. Blöbaum, A. A. Mastakouri, P. M. Faller, L. Minorics, and K. Budhathoki, "Quantifying intrinsic causal contributions via structure preserving interventions," 2024.

[4] J. Pearl, *Causality*. Cambridge University Press, 2 ed., 2009.

[5] A. Javaloy, P. S. Martin, and I. Valera, "Causal normalizing flows: from theory to practice," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[6] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp. 1530–1538, PMLR, 2015.

[7] M. J. Jansen, "Analysis of variance designs for model output," *Computer Physics Communications*, vol. 117, no. 1, pp. 35–43, 1999.

[8] G. Papamakarios, T. Pavlakou, and I. Murray, "Masked autoregressive flow for density estimation," 2018.

[9] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," in *JMLR*, 2012.

[10] V. Petsiuk, R. Jain, V. Manjunatha, V. I. Morariu, A. Mehra, V. Jain, and K. Saenko, "Beyond individualized feature attributions: A unified framework for faithful model interpretations," in *European Conference on Computer Vision (ECCV)*, pp. 309–328, Springer, 2022.

[11] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[12] Q. Xi and B. Bloem-Reddy, "Indeterminacy in generative models: Characterization and strong identifiability," in *International Conference on Artificial Intelligence and Statistics, 25-27 April 2023, Palau de Congressos, Valencia, Spain* (F. J. R. Ruiz, J. G. Dy, and J. van de Meent, eds.), vol. 206 of *Proceedings of Machine Learning Research*, pp. 6912–6939, PMLR, 2023.

[13] S. L. Lauritzen and D. J. Spiegelhalter, "Local computation with probabilities on graphical structures and their application to expert systems (with discussion)," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 50, no. 2, pp. 157–224, 1988.

[14] M. Scutari, "Bayesian network repository." https://www.bnlearn.com/bnrepository/discrete-small.html#cancer, 2023.

[15] H. Hofmann, "Statlog (German Credit Data)." UCI Machine Learning Repository, 1994. DOI: https://doi.org/10.24432/C5NC77.

[16] S. Saha and U. Garain, "On noise abduction for answering counterfactual queries: A practical outlook," *Transactions on Machine Learning Research*, 2022.

[17] A.-H. Karimi, B. Schölkopf, and I. Valera, "Algorithmic recourse: from counterfactual explanations to interventions," in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, (New York, NY, USA), p. 353–362, Association for Computing Machinery, 2021.

[18] W. Falcon, "Pytorch lightning." https://www.pytorchlightning.ai, 2020.

[19] A. Polyak, "Zuko: Flows in pytorch." https://zuko.readthedocs.io/, 2023.

[20] M. Ibrahim, M. Louie, C. Modarres, and J. Paisley, "Global explanations of neural networks: Mapping the landscape of predictions," in *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, (New York, NY, USA), p. 279–287, Association for Computing Machinery, 2019.

[21] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), p. 1135–1144, Association for Computing Machinery, 2016.

[22] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[23] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 3319–3328, PMLR, 2017.

[24] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 3145–3153, PMLR, 2017.

[25] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "SmoothGrad: Removing Noise by Adding Noise." https://arxiv.org/abs/1706.03825, 2017. arXiv preprint arXiv:1706.03825.

[26] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 4768–4777, Curran Associates Inc., 2017.

[27] C. Agarwal, S. Krishna, E. Saxena, M. Pawelczyk, N. Johnson, I. Puri, M. Zitnik, and H. Lakkaraju, "OpenXAI: Towards a transparent evaluation of model explanations," in *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[28] T. Fel, R. Cadene, M. Chalvidal, M. Cord, D. Vigouroux, and T. Serre, "Look at the variance! efficient black-box explanations with sobol-based sensitivity analysis," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[29] M. Gerber, "On integration methods based on scrambled nets of arbitrary size," *Journal of Complexity*, vol. 31, no. 6, pp. 798–816, 2015.

[30] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, "Top-down neural attention by excitation backprop," in *Proceedings of the European Conference on Computer Vision (ECCV)*, IEEE, 2016.

[31] V. Petsiuk, A. Das, and K. Saenko, "Rise: Randomized input sampling for explanation of black-box models," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

[32] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International Conference on Machine Learning*, 2017.

Appendix

5.1 Overview of Local Attribution Techniques

Local attribution methods are designed to identify how individual input features influence a model's prediction for a particular instance. Below, we summarize key approaches that are either gradient-based or perturbation-based.

5.1.1 Gradient × Input (I×G)

The Gradient × Input technique [24] highlights feature importance by multiplying each input feature with the gradient of the model output with respect to that feature. This is known to enhance the clarity of saliency maps. The attribution for input $\mathbf{x}$ is defined as:

$$a^{I\times G}(\mathbf{x}) = \mathbf{x} \odot |\nabla_{\mathbf{x}} f(\mathbf{x})|$$

5.1.2 Integrated Gradients (IG)

Integrated Gradients [32] estimate the contribution of each feature by integrating the gradient of the output along a straight-line path from a baseline input $\mathbf{x}_0$ (e.g., a zero vector) to the actual input $\mathbf{x}$. The formula is given by:

$$a^{IG}(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_0) \cdot \int_0^1 \nabla_{\mathbf{x}} f(\mathbf{x}_0 + \alpha(\mathbf{x} - \mathbf{x}_0))\, d\alpha$$

This integral is typically approximated numerically using a discrete sum with a fixed number of steps.

5.1.3 SmoothGrad (SG)

SmoothGrad [25] enhances gradient-based explanations by averaging the gradients from multiple noisy variants of the input. For an input $\mathbf{x}$, noise samples $\boldsymbol{\varepsilon}$ are drawn from a Gaussian distribution, and the attributions are computed as:

$$a^{SG}(\mathbf{x}) = \frac{1}{m} \sum_{i=1}^{m} \nabla_{\mathbf{x}} f(\mathbf{x} + \boldsymbol{\varepsilon}_i), \quad \boldsymbol{\varepsilon}_i \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

where m is the number of samples and σ is the noise standard deviation.

5.1.4 SHAP (Shapley Additive Explanations)

SHAP [26] uses concepts from game theory to assign importance values to features based on their marginal contributions to the model's prediction across all possible subsets of features. While exact computation is exponential in the number of features, practical implementations rely on approximations such as model-specific solvers (e.g., TreeSHAP for decision trees).

5.1.5 LIME (Local Interpretable Model-agnostic Explanations)

LIME [21] builds a simple interpretable model (often linear) around a prediction of interest by generating local perturbations of the input and weighting them based on proximity. This surrogate model mimics the black-box model's behavior in the local neighborhood, providing human-understandable explanations.