Extending Layer-wise Relevance Propagation in Neural Networks using Semiring Annotations

Antoine Groudiev
Computer Science Department,
École Normale Supérieure, PSL
Paris, France
antoine.groudiev@ens.psl.eu

Arkaprava Saha
Université Grenoble Alpes, CNRS,
Grenoble INP, LIG
Grenoble, France
arkaprava.saha@univ-grenoble-alpes.fr

Silviu Maniu
Université Grenoble Alpes, CNRS,
Grenoble INP, LIG
Grenoble, France
silviu.maniu@univ-grenoble-alpes.fr

Abstract

Recently, neural networks have allowed computers to solve numerous problems in various fields such as natural language processing and computer vision. Compared to traditional algorithms, machine learning models have proven to be both more successful and more difficult to interpret. Neural networks are considered as black boxes that are unable to easily explain themselves, i.e. justifying the reasons that led them to make a prediction. Layer-wise Relevance Propagation (LRP) is a technique that has been introduced to ensure explainability by identifying the input features relevant to the output choice. In parallel, research in provenance theory has developed annotation techniques, which can for instance be used to compute query provenance in databases.

In this paper, we extend LRP propagation rules to semiring-based provenance annotations of a network and implement semiring-based propagation rules for computer vision models of different scales. We show that different semirings lead to different types and granularities of explanations, and that such techniques can be applied to perform tasks like image mask computation and neural network pruning.

CCS Concepts

• **Computing methodologies** → **Neural networks**; **Algebraic algorithms**; • **Theory of computation** → **Data provenance**.

Keywords

Semiring, Provenance, Neural Network, Relevance

ACM Reference Format:
Antoine Groudiev, Arkaprava Saha, and Silviu Maniu. 2025. Extending Layer-wise Relevance Propagation in Neural Networks using Semiring Annotations. In *ProvenanceWeek (PW' 25), June 22–27, 2025, Berlin, Germany.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3736229.3736266

1 Introduction

Supervised machine learning is a long-studied field that helps in solving various real-world problems by using some data comprising inputs with given correct outputs (called training) to predict the output for completely new inputs (called testing). More recently, neural networks, which are well-known machine learning models, have been proven to be successful in solving such problems with high accuracy. Numerous variants of neural networks have been proposed for various applications, ranging from simple multilayer perceptrons (MLPs) for solving general classification and regression tasks to convolutional neural networks (CNNs) for problems on images, recurrent neural networks (RNNs) for problems on text and other sequential inputs, and transformers, which are used to create complex tools such as large language models (LLMs). The expressivity of the class of functions generated by neural networks, combined with the relative simplicity of their training, makes such models versatile tools for learning the relationship between the inputs and outputs of a dataset.

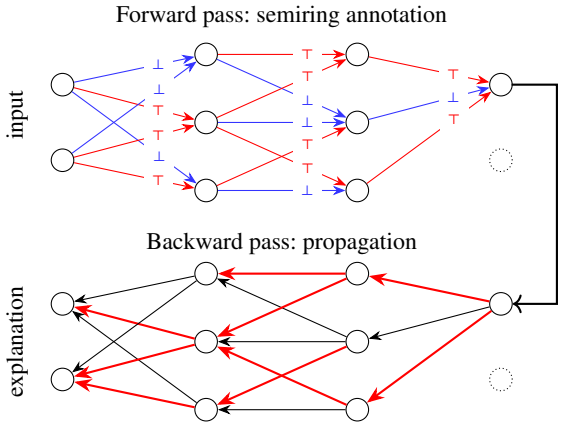

Figure 1: Illustration of the semiring-based layer-wise relevance propagation method. The forward pass annotates the neurons with semiring elements, while the backward pass propagates the relevance of the output neuron to the input neurons.

However, this versatility comes at the cost of poor interpretability: a neural network is simply a black box representing a function from one high-dimensional space to another, but provides no justification or explanation for a given execution. If metrics like accuracy over a test set provide confidence in the fact that the model is able to correctly classify inputs similar to the training set, no guarantee is given that the model generalizes well. Real-world examples show that networks can overfit training data or even take shortcuts instead of learning the intended solution [17]. For the end user to have confidence in its predictions, a neural network should therefore be able to highlight the patterns in the input data that it actually learned,

PW' 25, Berlin, Germany

ACM ISBN 979-8-4007-1941-7/2025/06
https://doi.org/10.1145/3736229.3736266

or more generally, provide an explanation of how it arrived at its conclusion. This is the motivation behind the rise of explainable artificial intelligence (XAI) [35].

Layer-wise Relevance Propagation (LRP) [6] has been introduced as a technique to explain a neural network's execution. LRP propagates the output of the function backward in the network, using various rules to compute the *relevance* of a neuron depending on the relevances of the neurons of the upstream layer. LRP introduces the notion of *relevance score* for a neuron, intuitively quantifying the contribution of this neuron to the final output. A high relevance score indicates that the neuron led to the activation of the considered output; a negative relevance score represents neurons that increased the activation of another output neuron instead of the one considered.

In parallel, the notion of data provenance in database theory has developed formal solutions to a similar problem. Data provenance aims at *explaining* a query by highlighting the tuples in the original database that lead to the presence of a certain tuple in the query result. Although contexts are different, explanation of deep neural networks and data provenance share the same general setup: identifying a subset of the input that directly implies a certain output, and showing how these were used to lead to the final answer.

A formal framework to approach data provenance is a *provenance semiring* [18], which annotates tuples using abstract elements of a semiring and applies semiring operations to the tuples appearing in the query. As the query is executed, information about the provenance of the intermediate results is aggregated, resulting in an abstract formula that can be instantiated by substituting abstract elements and operations by a concrete semiring. Similarly, in the context of graph databases, edges can be annotated to compute a variety of properties of a graph query result [30].

Contributions. In this paper, we take the first steps to extend the classic layer-wise relevance propagation using semiring annotations, as illustrated in Figure 1. While classic LRP models relevances of neurons as real numbers, our semiring-based LRP models them using abstract elements of a semiring. Also, for computing relevances of neurons in a layer, our model applies semiring operations (instead of the usual addition and multiplication) to those of neurons of the upper layer. As we show in our experiments, different semirings lead to different types and granularities of explanations, each of which can be useful for specific applications.

Our contributions are the following:

- We generalize LRP to annotation using semiring elements, with a goal to compute the relevance values of all neurons such that a certain conservation property (w.r.t. semiring operations) is satisfied.
- We propose an approximate solution to our problem using annotation functions which map real-valued neuron activations and weights to semiring elements. We formally define the annotation functions for the four semirings we study.
- We show experimentally, using two neural network architectures, that relevances based on different semirings provide varied explanation semantics in an image classification setup. We also demonstrate applications of semiring-based relevance to perform tasks such as computing image masks and pruning neural networks.

Outline. The remainder of this paper is organized as follows. In § 2 we review previous works on neural network explainability and data provenance. § 3 provides formal descriptions of neural networks, semirings and neuron relevance, followed by the formal statement of our problem. We propose our solution and analyze it in § 4. In § 5 we experimentally demonstrate the explanations provided by our semiring-based relevance, followed by applications to image mask computation and neural network pruning. § 6 concludes the paper and suggests some future research directions.

2 Related Work

This work is at the intersection of two research areas: explainability in neural networks and data provenance, as outlined below.

Explainability in Neural Networks. Given the inherent black-box nature of neural networks, explaining their behavior has been explored in a large number of works [35]. Some consider a neural network as a black box and do not access its internal structure [15], while others take advantage of the internal states and weights of a network [6, 11, 23, 31]. A distinction is also made between local and global explanations: a *local* result explains a single execution of the model, while a *global* result tries to understand the model for any possible execution [34]. Moreover, *self-explaining* approaches use only the data made available by the model computation during the prediction, while *post-hoc* approaches perform more operations after the initial inference. In addition, explanations of neural network outputs can take various forms. Mechanistic interpretability [7, 27] studies the fundamental components of networks through a granular analysis of features, neurons, layers, and connections, offering an intimate view of operational mechanics. Circuits [13, 20, 28], the minimal computational subgraphs of neural network models with behaviour faithful to that of the whole model for a given task, constitute one form of explanations offered by mechanistic interpretability. Causal interpretability [3, 26] is based on the abstraction of causal graphs, which represent causal relationships between various components of the model. Causal methods employ counterfactual interventions [22] on some part of the model or its input to identify the components that lead to the same output despite the interventions. Concept-based explanations [29] provide a more holistic view of the inner workings of the model by explaining its predictions in terms of human-understandable attributes or abstractions.

Data Provenance. Data provenance deals mainly with how one can track the origin of data through a process; in databases, that can be the origin of query outputs [9, 10]. Related to this is the concept of data lineage [2] and the corresponding Trio system. Theoretically, for SQL queries and database applications, the *provenance semiring* framework [18] has been shown to fully capture query provenance semantics; it also captures previous forms of provenance. Extensions of the semiring framework to more complex queries can be found in [4, 8, 16]. Examples of modern systems for tracking provenance in relational databases are ProvSQL [32] and GProM [5].

3 Preliminaries

In this paper we focus on neural networks trained for classification, where for input $\mathbf{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^d$, the output is one of c possible classes, with y being the ground-truth class for $\mathbf{x}$.

3.1 Neural Networks

A neural network is a set of layered units or nodes called neurons connected by edges between each pair of neurons in consecutive

layers. We denote the total number of layers as L and the neuron at position i in layer l as $v_i^{(l)}$, with the total number of neurons in layer l being n_l. Layer 1 corresponds to the input to the neural network and layer L to the output. The weights of edges from nodes in layer l to those in layer $l+1$ are represented as a matrix $W^{(l,l+1)} \in \mathbb{R}^{n_l \times n_{l+1}}$, with element $w_{ij}^{(l,l+1)}$ denoting the weight of the edge from node i in layer l to node j in layer $l+1$. Neuron i in layer l outputs a value (a.k.a. activation) $a_i^{(l)}$ defined by the following recurrence relation:

$$a_i^{(l)} = \begin{cases} x_i & \text{if } l = 1 \\ g^{(l)}\left(\sum_{j=0}^{n_{l-1}} a_j^{(l-1)} w_{ji}^{(l-1,l)}\right) & \text{otherwise} \end{cases} \quad (1)$$

Here $g^{(l)} : \mathbb{R}^{n_l} \to \mathbb{R}$ is a non-linear activation function (e.g. ReLU, tanh etc.). In order to incorporate node biases within this notation, we denote the bias of neuron j in layer $l+1$ as $w_{0j}^{(l,l+1)}$ and set $a_0^{(l)} = 1$ for $l \in [1, L]$. This is equivalent to inserting a dummy neuron with index 0 in each layer to serve the purpose of propagating bias to each neuron in the next layer.

3.2 Neuron Relevance and Semirings

In Layer-wise Relevance Propagation (LRP) [6] explanations are obtained by propagating the output relevance backwards through the network while respecting a conservation law. For the i-th neuron of layer l, LRP introduces the quantity $R_i^{(l)}$, called the *relevance* of the neuron, which quantifies how much this neuron contributed to the final classification. A simple way to distribute weights is

$$R_i^{(l)} = \begin{cases} a_i^{(l)} \times I[i = y] & \text{if } l = L \\ \left[\frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_{i'=0}^{n_l} a_{i'}^{(l)} w_{i'j}^{(l,l+1)}}\right]_{j=0}^{n_{l+1}} \times R_j^{(l+1)} & \text{otherwise} \end{cases} \quad (2)$$

Here $I[\cdot]$ is the indicator function (1 if true, 0 otherwise). For the last layer (L), only the neuron corresponding to the ground truth class y is relevant, and its relevance is equal to its activation. For all other layer, the relevance of a neuron i is the sum of the contributions of this neuron to the ones of the next layer, weighted by the quantity $a_i^{(l)} w_{ij}^{l,l+1}$, which models the extent to which neuron i contributes to the activation of neuron j in layer $l+1$. Note that if either the weight $w_{ij}^{(l,l+1)}$ or the activation $a_i^{(l)}$ is zero, then the neuron i does not contribute to the activation of neuron j in layer $l+1$, and hence its relevance is zero. The denominator is then added to ensure that the total relevance is conserved, i.e. the sum of the relevances of all neurons in layer l is equal to the sum of the relevances of all neurons in layer $l+1$.

We aim to extend Layer-wise Relevance Propagation by annotating the computational graph of a neural network with semiring elements. We begin by providing a mathematical definition of a monoid, using which we define a semiring.

Definition 1 (Monoid). *A monoid $(\mathbb{K}, \odot, e)$ is an algebraic structure composed of a set $\mathbb{K}$, a binary operator $\odot$ and an element $e \in \mathbb{K}$, satisfying the following properties:*

- $\forall a, b, c \in \mathbb{K},\ (a \odot b) \odot c = a \odot (b \odot c)$
- $\forall a \in \mathbb{K},\ a \odot e = e \odot a = a$

The monoid is called commutative if for all $a, b \in \mathbb{K}$, $a \odot b = b \odot a$. An element $o \in \mathbb{K}$ is called absorbing or annihilating if for all $a \in \mathbb{K}$, $o \odot a = a \odot o = o$.

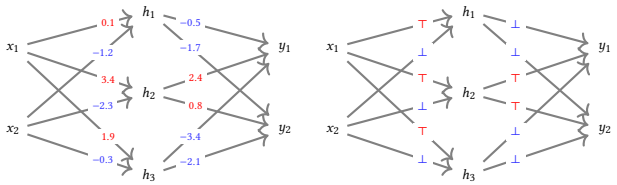

Figure 2: A simple neural network with edges annotated with real-valued weights (left) and boolean semiring elements (right).

Definition 2 (Semiring). *A semiring $(\mathbb{K}, \oplus, \otimes, \mathbb{0}, \mathbb{1})$ is an algebraic structure composed of a set $\mathbb{K}$, binary operators $\oplus$ and $\otimes$ such that $\otimes$ distributes over $\oplus$, satisfying the following properties:*

- *$(\mathbb{K}, \oplus, \mathbb{0})$ is a commutative monoid.*
- *$(\mathbb{K}, \otimes, \mathbb{1})$ is a monoid such that $\mathbb{0}$ is absorbing (annihilating).*

We provide four examples of semirings suited for application to deep neural networks.

Example 1 (Real semiring). *$(\mathbb{R}, +, \times, 0, 1)$ is a semiring. While it has no direct interpretation is the context of databases, we will see that it corresponds to the basic real-valued LRP.*

Example 2 (Boolean semiring). *$(\mathbb{B}, \vee, \wedge, \bot, \top)$ where $\mathbb{B} := \{\bot, \top\}$ is a semiring. In database provenance, its use is interpreted as the existence of a path between two vertices, using edge weights as the number of different paths between two adjacent vertices.*

Example 3 (Counting semiring). *$(\mathbb{N}, +, \times, 0, 1)$ is a semiring. For a non-cyclic graph database, its use allows to compute the total number of paths between two vertices, using edge weights as the number of different paths between two adjacent vertices.*

Example 4 (Viterbi semiring). *$([0, 1], \max, \times, 0, 1)$ is a semiring. For a non-cyclic graph database where the annotations are interpreted as a confidence measure, its use allows to compute the confidence score of the result of a query.*

Similar to the classic LRP [6], we can express semiring layer-wise relevance in terms of messages sent to neurons of the previous layers. We denote by $R_{i \leftarrow j}^{(l,l+1)}$ the message received by neuron i in layer l from neuron j in layer $l+1$. Note that the messages are directed from a neuron towards its input neurons, in contrast to what happens at prediction time. Specifically, we define the relevance of a neuron as the sum (w.r.t. the addition operator of the semiring) of its incoming messages, except for those in the output layer whose relevance is simply whether or not they correspond to the ground truth class y for the input $\mathbf{x}$. This definition is formalized below.

Definition 3 (Neuron Relevance). *The relevance of neuron i in layer l w.r.t. semiring $(\mathbb{K}, \oplus, \otimes, \mathbb{0}, \mathbb{1})$ is defined as*

$$R_i^{(l)} = \begin{cases} \mathbb{1}[i = y] & \text{if } l = L \\ \bigoplus_{j=0}^{n_{l+1}} R_{i \leftarrow j}^{(l,l+1)} & \text{otherwise} \end{cases} \quad (3)$$

where $\mathbb{1}[\cdot]$ is the indicator function ($\mathbb{1}$ if true, $\mathbb{0}$ otherwise).

Intuitively, (3) can be interpreted in two ways, as illustrated in Figure 2. Firstly, it can be seen as the abstraction of an LRP computation using formal semiring elements. Computing relevance w.r.t. $(\mathbb{K}, \oplus, \otimes, \mathbb{0}, \mathbb{1})$ for an execution of the network results in an abstract formula in terms of elements of $\mathbb{K}$ and operations $\oplus$ and $\otimes$. While

this might be useful in the context of graph provenance, visualizing abstract formulae is difficult because of the size of the computational graphs associated with neural networks. Furthermore, the general structure of the formulae is always the same for a specific model, since neural networks usually have very simple computational graph structures. Therefore, a second approach that fits the graph properties of neural networks is to interpret (3) as operations on an annotated circuit, similar to [32]. For instance, in the case of the boolean semiring, computing relevance consists of taking, for each node, the logical conjunction of the annotations of its outgoing edges.

3.3 Problem Statement

We would like to find message values (and hence neuron relevances) which ensure that the overall relevance of all neurons is conserved (remains the same) in each layer. An even stronger condition is that the sum (w.r.t. the addition operator of the semiring) of the messages entering a neuron, which is the relevance of the neuron, is equal to the sum of the messages leaving it. We formally state this problem.

PROBLEM 1. *Given a trained neural network with L layers and a semiring $(\mathbb{K}, \oplus, \otimes, \mathbb{0}, \mathbb{1})$, find a set of messages*

$$\left\{R_{i\leftarrow j}^{(l,l+1)} \in \mathbb{K} : i \in [0, n_l], j \in [0, n_{l+1}], l \in [1, L-1]\right\} \tag{4}$$

such that for each $l \in [2, L]$ and $j \in [0, n_l]$

$$\bigoplus_{i=0}^{n_{l-1}} R_{i\leftarrow j}^{(l-1,l)} = R_j^{(l)} \tag{5}$$

The difference between (5) and (3) is that in (5) the sum runs over the "sources" at layer l for a fixed neuron j at layer $l+1$, while in (3) the sum runs over the 'sinks' at layer $l+1$ for a fixed neuron i at a layer l. One can interpret (5) by saying that the messages $R_{i\leftarrow j}^{(l-1,l)}$ are used to distribute the relevance of a neuron j to its input neurons in layer l.

4 Solution

We propose an approximate solution to Problem 1 and analyze its quality and running time, followed by instantiations on the four semirings in §3.2.

4.1 Algorithm and Analysis

As mentioned in §3.2, knowing the relevance of a certain neuron for the classification decision, we would like to obtain a decomposition of the relevance in terms of messages sent to neurons of the previous layers. Suppose we are given a semiring $(\mathbb{K}, \oplus, \otimes, \mathbb{0}, \mathbb{1})$. An obvious choice would be to assign the values of the messages proportional to the activations a neuron receives from those in its previous layer. However, notice that neuron activations and edge weights are real numbers, whereas our relevance values must be from $\mathbb{K}$. Thus, for all layers of the neural network, we define annotation functions $\left\{\Theta^{(l)} : l \in [1, L-1]\right\}$ which map real numbers to values in $\mathbb{K}$. Given this, we set the values of the message sent by a neuron j in layer $l+1$ to neuron i in layer l to $\Theta^{(l)}(p)$, where p is proportional to the activation that i sends j. That is,

$$R_{i\leftarrow j}^{(l,l+1)} = \Theta^{(l)}\left(\frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_{i'=0}^{n_l} a_{i'}^{(l)} w_{i'j}^{(l,l+1)}}\right) \otimes R_j^{(l+1)} \tag{6}$$

Algorithm 1 Find Messages

Input: A trained neural network with L layers, input vector $\mathbf{x}$ classified as output class y, a semiring $(\mathbb{K}, \oplus, \otimes, \mathbb{0}, \mathbb{1})$, annotation functions $\left\{\Theta^{(l)} : l \in [1, L-1]\right\}$

Output: $\left\{R_{i\leftarrow j}^{(l,l+1)} \in \mathbb{K} : i \in [0, n_l], j \in [0, n_{l+1}], l \in [1, L-1]\right\}$

1: **for** $i = 0$ **to** n_L **do**
2: $\quad R_i^{(L)} \leftarrow \mathbb{1}[i = y]$
3: **for** $l = L-1$ **to** 1 **do**
4: $\quad$ **for** $i = 0$ **to** n_l **do**
5: $\quad\quad$ **for** $j = 0$ **to** n_{l+1} **do**
6: $\quad\quad\quad R_{i\leftarrow j}^{(l,l+1)} \leftarrow \Theta^{(l)}\left(\frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_{i'=0}^{n_l} a_{i'}^{(l)} w_{i'j}^{(l,l+1)}}\right) \otimes R_j^{(l+1)}$
7: $\quad\quad R_i^{(l)} \leftarrow \bigoplus_{j=0}^{n_{l+1}} R_{i\leftarrow j}^{(l,l+1)}$
8: **return** $\left\{R_{i\leftarrow j}^{(l,l+1)} : i \in [0, n_l], j \in [0, n_{l+1}], l \in [1, L-1]\right\}$

Once these messages are computed, the overall relevance of a neuron in layer l is determined by adding up the messages coming from all neurons in layer $l+1$, in accordance with (3). This means

$$R_i^{(l)} = \bigoplus_{j=0}^{n_{l+1}} R_{i\leftarrow j}^{(l,l+1)} = \left[\bigoplus_{j=0}^{n_{l+1}} \Theta^{(l)}\left(\frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_{i'=0}^{n_l} a_{i'}^{(l)} w_{i'j}^{(l,l+1)}}\right)\right] \otimes R_j^{(l+1)} \tag{7}$$

The pseudocode for the whole method is shown in Algorithm 1.

We now analyze our solution. The relevances computed by our algorithm can be shown to satisfy the following.

THEOREM 1. *Algorithm 1 returns messages $R_{i\leftarrow j}^{(l,l+1)}$ such that for each $l \in [2, L]$ and $j \in [0, n_l]$*

$$\bigoplus_{i=0}^{n_{l-1}} R_{i\leftarrow j}^{(l-1,l)} = \left[\bigoplus_{i=0}^{n_{l-1}} \Theta^{(l-1)}\left(\frac{a_i^{(l-1)} w_{ij}^{(l-1,l)}}{\sum_{i'=0}^{n_{l-1}} a_{i'}^{(l-1)} w_{i'j}^{(l-1,l)}}\right)\right] \otimes R_j^{(l)} \tag{8}$$

PROOF. Replacing l with $l-1$ in (6) and applying $\oplus$ across all possible values of i (0 to n_{l-1}), we get the result. □

Note that (8) is a generalized approximation to (5), in that the right-hand side of (8) is a certain quantity "multiplied" ($\otimes$ instead of $\times$) by $R_j^{(l)}$, in contrast to just $R_j^{(l)}$ on the right-hand side of (5).

Moreover, the worst-case running time of our method can be derived as follows.

THEOREM 2. *Algorithm 1 has a time complexity of $O\left(Ln_m^2\right)$, where n_m is the maximum number of neurons in a layer.*

PROOF. The majority of the time is spent on the iterative computations of line 6. The sum in the denominator can be precomputed just after line 3; thus, each computation of line 6 takes constant time. Hence, the total running time is $O\left(\sum_{l=1}^{L-1} n_l n_{l+1}\right)$ or $O\left(Ln_m^2\right)$. □

4.2 Examples of Semirings

As mentioned above, our relevance values can be computed given annotation functions which map real weighted activation values to semiring values. The exact definitions of such activation functions, however, depend on the semiring under consideration. In this section, we define the activation functions for the semirings in §3.2.

Real Semiring. For the real semiring $(\mathbb{R}, +, \times, 0, 1)$, the relevance can be computed by setting the annotation function in all layers

equal to the identity function, i.e. $\Theta : \mathbb{R} \to \mathbb{R}$ defined as $\Theta(x) = x$. Thus, the expression for the relevance of a node i in layer l in terms of those in layer $l+1$ becomes

$$R_i^{(l)} = \left[\sum_{j=0}^{n_{l+1}} \frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_{i'=0}^{n_l} a_{i'}^{(l)} w_{i'j}^{(l,l+1)}}\right] \times R_j^{(l+1)} \tag{9}$$

Note that (9) combined with (3) is identical to (2) upto a multiplicative factor of $a_i^{(l)}$ because this factor appears, for the case $i = L$ and hence also propagated to the other layers, in (2) but not in (3).

Boolean Semiring. For the boolean semiring $(\mathbb{B}, \vee, \wedge, \bot, \top)$ where $\mathbb{B} := \{\bot, \top\}$, the relevance (also called $\mathbb{B}$-relevance) can be computed using the same annotation function $\Theta : \mathbb{R} \to \mathbb{B}$ for all layers, defined as $\Theta(x) = \mathbb{1}[x \geq \theta]$, where θ is a hyperparameter called the threshold. Thus, the expression for the relevance of a node i in layer l in terms of those in layer $l+1$ becomes

$$R_i^{(l)} = \left[\bigvee_{j=0}^{n_{l+1}} \mathbb{1}\left(\frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_{i'=0}^{n_l} a_{i'}^{(l)} w_{i'j}^{(l,l+1)}} \geq \theta\right)\right] \wedge R_j^{(l+1)} \tag{10}$$

Counting Semiring. For the counting semiring $(\mathbb{N}, +, \times, 0, 1)$, the relevance can be computed using the same annotation function $\Theta : \mathbb{R} \to \mathbb{N}$ for all layers, defined similarly to those for the boolean semiring, i.e. $\Theta(x) = \mathbb{1}[x \geq \theta]$, where θ is a hyperparameter called the threshold. Thus, the expression for the relevance of a node i in layer l in terms of those in layer $l+1$ becomes

$$R_i^{(l)} = \left[\sum_{j=0}^{n_{l+1}} \mathbb{1}\left(\frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_{i'=0}^{n_l} a_{i'}^{(l)} w_{i'j}^{(l,l+1)}} \geq \theta\right)\right] \times R_j^{(l+1)} \tag{11}$$

Viterbi Semiring. For the Viterbi semiring $([0,1], \max, \times, 0, 1)$, the relevance can be computed using the annotation function $\Theta : \mathbb{R} \to [0,1]$ defined as

$$\Theta\left(\frac{p_i}{\sum_{i'} p_{i'}}\right) = \frac{|p_i|}{\max_{i'} |p_{i'}|} \tag{12}$$

guaranteeing the conservation property (5). Thus, the expression for the relevance of a node i in layer l in terms of those in layer $l+1$ becomes

$$R_i^{(l)} = \left[\max_{j=0}^{n_{l+1}}\left(\frac{\left|a_i^{(l)} w_{ij}^{(l,l+1)}\right|}{\max_{i'=0}^{n_l}\left|a_{i'}^{(l)} w_{i'j}^{(l,l+1)}\right|}\right)\right] \times R_j^{(l+1)} \tag{13}$$

5 Experiments

We run experiments to demonstrate the types of explanations generated by using different semirings (§ 5.2 and § 5.3), along with applications of semiring-based LRP to tasks like image mask computation (§ 5.4) and neural network pruning (§ 5.5). The source code [19] is written in Python and run on a 3.22 GHz macOS laptop with 32 GB RAM and an integrated 14-Core-GPU.

5.1 Setup

Neural networks and datasets. We use two different neural networks, each trained on different datasets, as used in [6, 25]. The first is a fully connected rectifier neural network with 4 layers of sizes 784, 300, 100 and 10, trained on the MNIST handwritten digits dataset [21]. The second is the deep convolutional network VGG-16 [33] trained over the ImageNet visual dataset [14]. While such examples are not representative of the state-of-the-art in deep learning, they are simple enough to allow for a clear understanding of the relevance propagation process. Layer-wise relevance propagation has been shown to be scalable to the latest architectures, such as transformer models [1], and we believe that the same principles apply to our semiring-based LRP. We let the generalization of semiring-based LRP to more complex architectures as future work.

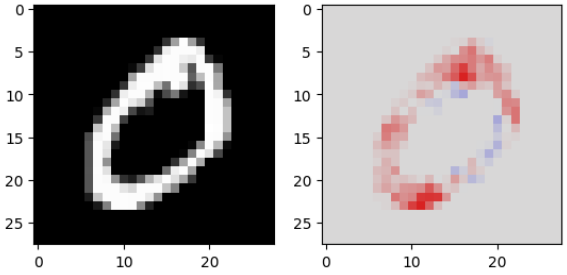

Figure 3: Input image and real-valued relevance (standard LRP) for the output neuron `0`.

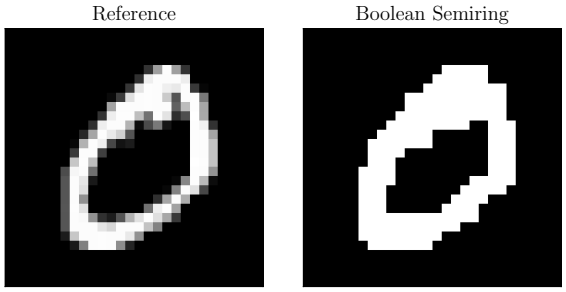

Figure 4: Input image and Boolean relevance for the output neuron `0` for $\theta = 10^{-9}$.

Semirings. We compare the explanations given by the real, boolean, counting and Viterbi semirings (§ 3.2).

5.2 Results on the MNIST Dataset

Figure 3 shows relevance (real semiring) for the fully connected rectifier network trained on the MNIST dataset. Pixels highlighted in red have a positive relevance while blue pixels have a negative relevance. As expected, the relevant pixels to classify this image as a `0` are the white pixels in the input image.

As shown in Figure 4, relevance with the Boolean semiring provides a higher level explanation, highlighting large zones of the input image which contribute the most to the final classification. Naturally, information about nuances in the contribution is lost. Intuitively, activated pixels (input neurons with relevance $\top$) are neurons such that there exists a relevant path from this neuron to the output neuron of the class `0`. A relevant path is a path in which all edges have a weight higher than the threshold θ. While this condition might seem quite restrictive, note that computation graphs for neural networks are densely connected: there is $784 \times 300 \times 100 = 23\,520\,000$ different paths connecting one input pixel to the output neuron of the class `0`.

As far as the counting semiring is concerned, its element-wise annotation function Θ is mostly identical to that for the Boolean semiring, but the operators + and $\times$ bring more expressivity to the framework, bringing more nuance in the highlighted zones. For small networks like the one studied, the number of relevant paths from

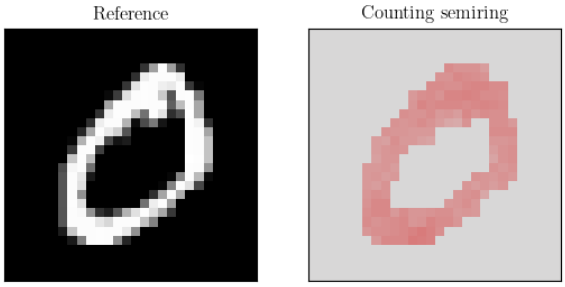

Figure 5: Input image and counting relevance for the output neuron `0` for $\theta = 10^{-9}$. Highest relevance is 2098.

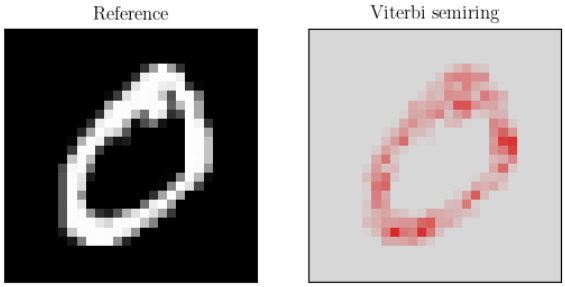

Figure 6: Input image and Viterbi relevance for the output neuron `0`.

Figure 7: Input image and relevance for the class `"castle"` of the VGG-16 network.

input to output neurons appears to be roughly the same, making the final result (Figure 5) qualitatively similar to Figure 4.

The Viterbi semiring replaces the + operator of the real and counting semirings by a max operator. In a context where neurons can have the same weighted sum of relevance, taking the maximum of the relevance emphasizes the most important neurons, giving a more contrasted visualization than classical LRP, as shown in Figure 6.

5.3 Results on the VGG-16 Network

Figure 7 shows a visualization of the relevance (real semiring) for the VGG-16 network over an image of the class `"castle"`. Note that the castle part of the image is highlighted in red as intended. Furthermore, both the street sign and the street light have strong negative relevance; those two objects correspond to other classes of the ImageNet dataset (`street sign (919)` and `traffic light (920)`). Those two elements of the image would have positive relevance for LRP starting from the output neurons `919` and `920`.

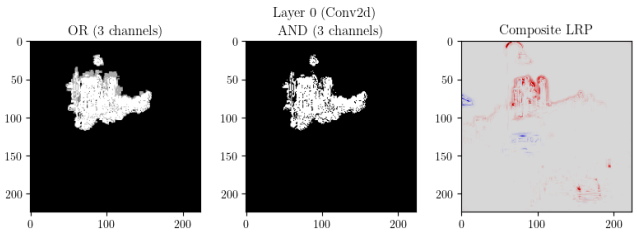

Figure 8: Boolean relevance of the class `castle`.

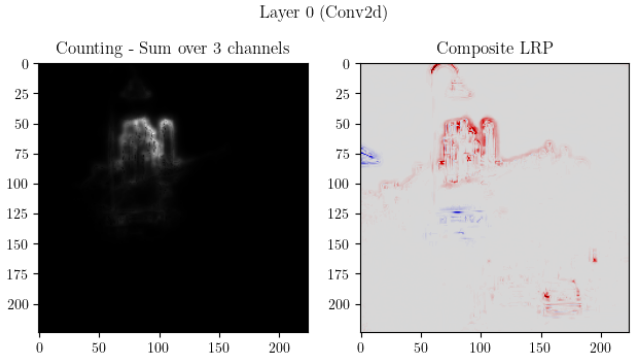

Figure 9: Counting relevance of the class `castle`.

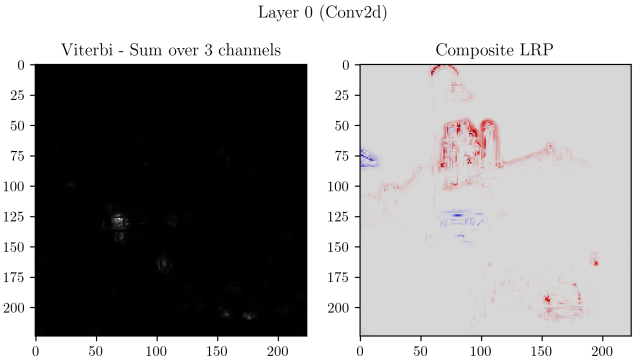

Figure 10: Viterbi relevance of the class `castle` for layer 0.

Boolean relevance over VGG-16 (Figure 8) is coherent with results on the MNIST dataset: it specifies a main zone of importance in the image, similarly to explanations of tools such as Grad-CAM [31]. The three channels of the input layer (R, G, B) are aggregated using OR/AND operations. While classical LRP provides a more precise explanation by focusing on the contours of the relevant object, Boolean relevance selects all the pixels of the object, including its center. From a human point of view, there is no reason to believe that the contours of the object are more relevant than the center of the object. Therefore, Boolean LRP allows for a larger explanation that is closer to what a human would produce.

The counting semiring (Figure 9) produces an explanation qualitatively similar to classical LRP. Compared to classic LRP, the choice of thresholds limits the propagation of relevance to provide a more localized explanation. This avoids the inclusion of less relevant elements such as the street light on top of the image.

The Viterbi semiring relevance appears to scale poorly to deep convolutional neural networks (Figure 10). Compared to the counting

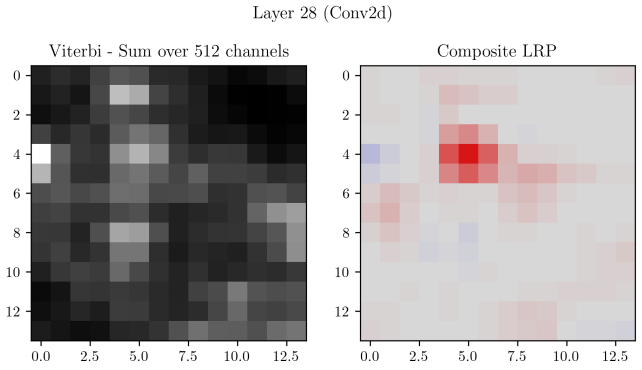

Figure 11: Viterbi relevance of the class `castle` for the layer 28.

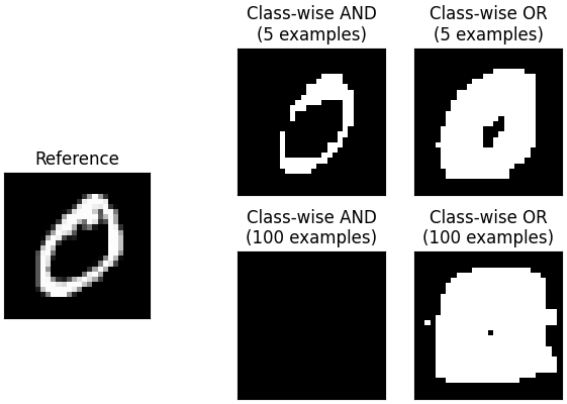

Figure 12: Class-wise mask for Boolean relevance of the class `0`.

or boolean semirings, it does not use thresholds to limit the spread of relevance during convolutions, which leads to the uniformization of relevance throughout the image. Therefore, the explanation obtained on the input layer is not humanly interpretable, as it concentrated on a very small number of pixels that are not even relevant to the classification. Nevertheless, an interesting aspect of CNNs is that the intermediate LRP results are also visualizable, since they respect spatial information. We can therefore display the relevance for a deep layer (in Figure 11, the 28th layer) to understand why the relevance propagation failed with this semiring.[1] In this case, we can see that even at the beginning of the propagation, the relevance is already spread out in the image. Information about the specific zone of the castle is already lost, and cannot be recovered by the lowest layers.

5.4 Image Mask Computation

A simple application consists of computing a mask by combining multiple relevance results. For instance, Figure 12 shows how the Boolean relevances of multiple executions from the same class can be combined using boolean operators to build image masks. These masks delimitate spatial regions of the image that most likely contain meaningful features used to classify images as the class `0`. For instance, the most useful input pixels for the classification as `0` are spatially located in a ring around the center of the image.

The same method also provides information on the internal neurons of the network that were used in the final classification. We believe that this technique can be used in the identification of *circuits* inside the neural network [11], that is computational subgraphs of the network mostly responsible for one specific behavior of the model.

The spatial localization of relevant pixels for a certain class do not scale to more complex models, where the dataset is not centered: if a certain object can be detected anywhere in the input image, the final mask will capture the entire image. Nevertheless, deeper neurons of networks are believed to be mostly spatially invariant, and mask computation can therefore be applied to the deepest layers.

To demonstrate the potential of mask computation using LRP, we built a toy example based on MNIST. We generated a new dataset in which images are made of two MNIST digits, one above the other

[1]Note that it remains harder to interpret deeper layers because of the way that channels are aggregated: here, we always fused channels by summing them, but this approach works better for 3 channels (input layer) than for 512 channels (28th layer).

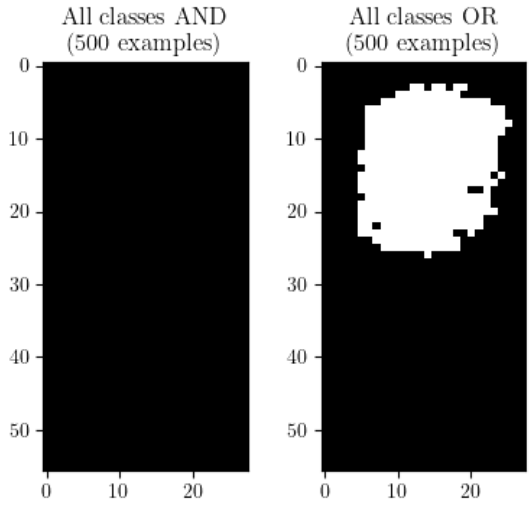

Figure 13: Mask of the relevant part of the images, obtained by applying $\wedge$ and $\vee$ pixel-wise to the results of boolean LRP.

(see Figure 14). The label for each image is the uppermost digit; the lower digit has no influence over the label and is used simply to confuse the classifier. We expect the relevance of the lower part of the image to be null. Figure 13 computes a mask of the relevant part of the images by combining multiple Boolean LRP results. Without prior knowledge of the connection between the label and the digits in the images, we can see that the model is learning to detect the upper digit in the image, and discards the lower digit.

The same procedure can be applied only for one class at a time, like in Figure 14.

5.5 Neural Network Pruning using Relevances

The accuracy of modern neural networks often come at the cost of large model size, increasing both storage, computational cost and inference time. To reduce the size of such models, an efficient technique called *network pruning* has been studied [12, 24], in which neurons are removed from the network. A simple approach to selecting which neurons to prune is to select those with the smallest ℓ_1 or ℓ_2 norm: that is, those whose associated column in the weight matrix has the smallest norm. While this selection method yields good

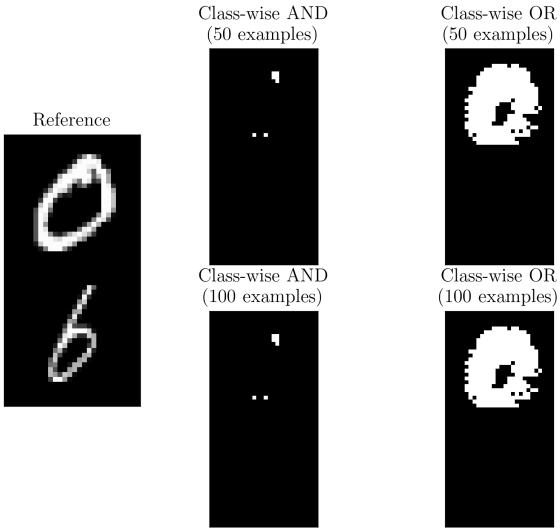

Figure 14: Computation of the image mask, only for images classified as `0`.

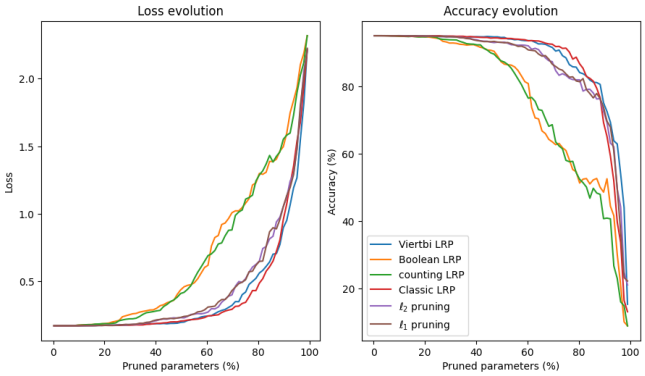

Figure 15: Accuracy and loss evolution during network pruning for different neuron selection methods.

results, we show that removing neurons with the smallest average relevance – in the sense of LRP – allows the network to perform better.

Figure 15 compares multiple selection methods. For each method, the network was evaluated multiple times on a subset of the MNIST test set, each time with an increasing number of neurons pruned from the first layer, selected using the specific method. For LRP-based methods, the relevance of each neuron is computed by taking its average relevance on LRP results over images of the training set, and neurons with the lowest absolute relevance are pruned first.

Three curve clusters can be identified: boolean and counting LRP perform much worse than other selection methods, which can be explained by the lack of granularity in the explanation that these techniques provide. The ℓ_1 and ℓ_2 norm selection methods provide a baseline to compare LRP techniques to, and both perform similarly well. Finally, classic LRP and Viterbi LRP outperform the norm-based methods by a small amount: this confirms the intuition that

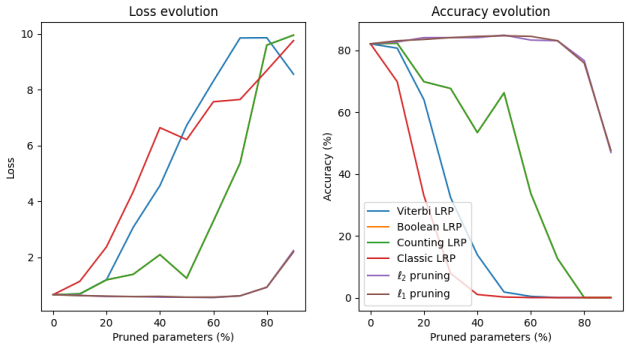

Figure 16: Accuracy and loss evolution during network pruning for different neuron selection methods.

LRP provides a more subtle explanation than simply the magnitude of the weights connected to a neuron.

We also tried to extend network pruning using relevance values to CNNs. We pruned VGG-16 using the same selection methods as MNIST, with relevances computed on a 224×224 image of class "`castle`"; thus, we evaluated the network on about 500 other images of the same class resized (if necessary) to 224×224. The results, shown in Figure 16, reveal nearly opposite trends to those of MNIST; the ℓ_2 and ℓ_1 norm selection methods perform better. This implies that we may need to incorporate more details in our method for CNNs, e.g. designing a method to compute the optimal values of the thresholds for the semirings (§ 4.2). This is an interesting direction for future work.

6 Conclusion

We introduced Semiring-based Layer-wise Relevance Propagation to extend LRP to a broader range of explanations. Semiring-based LRP benefits from the capabilities of classic LRP, while letting the user choose customized semirings for explanations that better suit some applications or are easier to intepret. Experimentally, the boolean and counting semirings provide scalable explanations that qualitatively differ from classical LRP. We show that the semiring generalization of LRP encapsulates different explanations types that go beyond classical LRP. In the future, for usefulness to larger-scale applications, it would be interesting to study other types of layers in this framework: we proved that semiring-based LRP could scale to CNNs, but architectures such as RNNs or transformers could also benefit. While we experimentally focused on image classification tasks, we also believe that semiring-based LRP can be applied to other deep learning problems, such as natural language processing. Moreover, scalability raises the challenge of performance, as the current high-level implementation would be too slow for most applications. Lower-level `PyTorch` modifications to allow semiring provenance would most likely result in improved performance.

Acknowledgments

This work was supported by a French government France 2030 program grant managed by the Agence Nationale de la Recherche (ANR), reference ANR-23-IACL-0006.

References

[1] Reduan Achtibat, Sayed Mohammad Vakilzadeh Hatefi, Maximilian Dreyer, Aakriti Jain, Thomas Wiegand, Sebastian Lapuschkin, and Wojciech Samek. 2024. AttnLRP: Attention-Aware Layer-Wise Relevance Propagation for Transformers. arXiv:2402.05602 [cs.CL]

[2] Parag Agrawal, Omar Benjelloun, Anish Das Sarma, Chris Hayworth, Shubha U. Nabar, Tomoe Sugihara, and Jennifer Widom. 2006. Trio: A System for Data, Uncertainty, and Lineage. In *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*, Umeshwar Dayal, Kyu-Young Whang, David B. Lomet, Gustavo Alonso, Guy M. Lohman, Martin L. Kersten, Sang Kyun Cha, and Young-Kuk Kim (Eds.). ACM, 1151–1154.

[3] Ola Ahmad, Nicolas Béreux, Loïc Baret, Vahid Hashemi, and Freddy Lecue. 2024. Causal Analysis for Robust Iterpretability of Neural Networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 4685–4694.

[4] Yael Amsterdamer, Daniel Deutch, and Val Tannen. 2011. Provenance for Aggregate Queries. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Maurizio Lenzerini and Thomas Schwentick (Eds.). ACM, 153–164.

[5] Bahareh Sadat Arab, Su Feng, Boris Glavic, Seokki Lee, Xing Niu, and Qitian Zeng. 2018. GProM - A Swiss Army Knife for Your Provenance Needs. *IEEE Data Eng. Bull.* 41, 1 (2018), 51–62.

[6] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE* 10, 7 (2015), 1–46.

[7] Leonard Bereska and Efstratios Gavves. 2024. Mechanistic Interpretability for AI Safety–A Review. *Transactions on Machine Learning Research (TMLR)* (2024), 55 pages.

[8] Pierre Bourhis, Daniel Deutch, and Yuval Moskovitch. 2020. Equivalence-Invariant Algebraic Provenance for Hyperplane Update Queries. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, David Maier, Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo (Eds.). ACM, 415–429.

[9] Peter Buneman, Sanjeev Khanna, and Wang Chiew Tan. 2000. Data Provenance: Some Basic Issues. In *Foundations of Software Technology and Theoretical Computer Science, 20th Conference, FST TCS 2000 New Delhi, India, December 13-15, 2000, Proceedings (Lecture Notes in Computer Science, Vol. 1974)*, Sanjiv Kapoor and Sanjiva Prasad (Eds.). Springer, 87–93.

[10] Peter Buneman, Sanjeev Khanna, and Wang Chiew Tan. 2001. Why and Where: A Characterization of Data Provenance. In *ICDT 2001 - 8th International Conference on Database Theory (Lecture Notes in Computer Science, Vol. 1973)*, Jan Van den Bussche and Victor Vianu (Eds.). Springer, 316–330.

[11] Nick Cammarata, Shan Carter, Gabriel Goh, Chris Olah, Michael Petrov, Ludwig Schubert, Chelsea Voss, Ben Egan, and Swee Kiat Lim. 2020. Thread: Circuits. *Distill* (2020).

[12] Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. 2024. A Survey on Deep Neural Network Pruning: Taxonomy, Comparison, Analysis, and Recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46, 12 (2024), 10558–10578.

[13] Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards Automated Circuit Discovery for Mechanistic Interpretability. *Advances in Neural Information Processing Systems* 36 (2023), 16318–16352.

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A Large-scale Hierarchical Image Database. In *2009 IEEE conference on computer vision and pattern recognition*. 248–255.

[15] Ruth C Fong and Andrea Vedaldi. 2017. Interpretable Explanations of Black Boxes by Meaningful Perturbation. In *Proceedings of the IEEE international conference on computer vision*. 3429–3437.

[16] Floris Geerts and Antonella Poggi. 2010. On Database Query Languages for K-relations. *J. Appl. Log.* 8, 2 (2010), 173–185.

[17] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut Learning in Deep Neural Networks. *Nature Machine Intelligence* 2 (2020), 665–673.

[18] Todd J Green, Grigoris Karvounarakis, and Val Tannen. 2007. Provenance semirings. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 31–40.

[19] Antoine Groudiev, Arkaprava Saha, and Silviu Maniu. 2025. Extending Layer-wise Relevance Propagation in Neural Networks using Semiring Annotations: Code Repository. https://github.com/Red-Rapious/Semiring-LRP.

[20] Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does GPT-2 Compute Greater-than?: Interpreting Mathematical Abilities in a Pre-Trained Language Model. *Advances in Neural Information Processing Systems* 36 (2023), 76033–76060.

[21] Yann LeCun. 1998. The MNIST Database of Handwritten Digits. http://yann.lecun.com/exdb/mnist/

[22] David Lewis. 2013. *Counterfactuals*. John Wiley & Sons.

[23] Aravindh Mahendran and Andrea Vedaldi. 2015. Understanding Deep Image Representations by Inverting Them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5188–5196.

[24] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2017. Pruning Convolutional Neural Networks for Resource Efficient Inference. In *5th International Conference on Learning Representations*. 30–46.

[25] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. 2019. *Layer-Wise Relevance Propagation: An Overview*. Springer International Publishing, 193–209.

[26] Aaron Mueller, Jannik Brinkmann, Millicent Li, Samuel Marks, Koyena Pal, Nikhil Prakash, Can Rager, Aruna Sankaranarayanan, Arnab Sen Sharma, Jiuding Sun, et al. 2024. The Quest for the Right Mediator: A History, Survey, and Theoretical Grounding of Causal Interpretability. *arXiv preprint arXiv:2408.01416* (2024), 36 pages.

[27] Neel Nanda, Andrew Lee, and Martin Wattenberg. 2023. Emergent Linear Representations in World Models of Self-Supervised Sequence Models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*. 16–30.

[28] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom In: An Introduction to Circuits. *Distill* 5, 3 (2020), e00024–001.

[29] Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. 2023. Concept-based Explainable Artificial Antelligence: A Survey. *arXiv preprint arXiv:2312.12936* (2023).

[30] Yann Ramusat, Silviu Maniu, and Pierre Senellart. 2021. Provenance-Based Algorithms for Rich Queries over Graph Databases. In *EDBT 2021 - 24th International Conference on Extending Database Technology*. 73–84. https://inria.hal.science/hal-03140067

[31] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. In *Proceedings of the IEEE international conference on computer vision*. 618–626.

[32] Pierre Senellart. 2018. Provenance and Probabilities in Relational Databases. *ACM SIGMOD Record* 46, 4 (2018), 5–15.

[33] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. https://arxiv.org/abs/1409.1556

[34] Matthew D Zeiler and Rob Fergus. 2014. Visualizing and Understanding Convolutional Networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*. Springer, 818–833.

[35] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. 2021. A Survey on Neural Network Interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence* 5, 5 (2021), 726–742.