Rule Extraction in Trained Feedforward Deep Neural Networks - Integrating Cosine Similarity and Logic for Explainability

» **Pablo Negro**
CAETI – Centro de Altos Estudios en tecnología informática, Facultad de Tecnología Informática. UAI.

Claudia Pons
CAETI – Centro de Altos Estudios en tecnología informática, Facultad de Tecnología Informática. UAI.
LIFIA – Facultad de Informática. Universidad Nacional de La Plata.
CIC – Comisión de Investigaciones Científicas de Buenos Aires CIC-PBA.

Received: 2024 September./ Accepted: 2024 November / Published: 2024 December.
How to Cite: Negro P, Pons C. Rule Extraction in Trained Feedforward Deep Neural Networks - Integrating Cosine Similarity and Logic for Explainability. Revista Abierta de Informática Aplicada [Internet]. [cited 2025 Jan. 20];8(1):9-34. https://doi.org/10.59471/raia2024203

Abstract

Explainability is a fundamental aspect in the field of machine learning, particularly in ensuring transparency and trust in decision-making processes.
As the complexity of machine learning models increases, the integration of neural and symbolic approaches has emerged as a promising solution to the explainability problem. In this context, the utilization of search methods for rule extraction in trained deep neural networks has been proven effective.
This involves the examination of weight and bias values generated by the network, typically through calculating the correlation between weight vectors and outputs. The hypothesis developed in this article states that by incorporating cosine similarity in this process, the search space can be efficiently narrowed down to identify the critical path connecting inputs to results. Furthermore, to provide a more comprehensive and interpretable understanding of the decision-making process, this article proposes the integration of first-order logic (FOL) in the rule extraction process. By leveraging cosine similarity and FOL, a groundbreaking algorithm that is capable of extracting and explaining the rule patterns learned by a feedforward trained neural network was designed and implemented. The algorithm was tested in three use cases showing effectiveness in providing insights into the model's behavior.

KEYWORDS: Artificial Intelligence, Cosine Similarity, Deep Learning, Explainability, Logic, Rule Extraction

Resumen

La explicabilidad es un aspecto fundamental en el campo del aprendizaje automático, en particular para garantizar la transparencia y la confianza en los procesos de toma de decisiones. A medida que aumenta la complejidad de los modelos de aprendizaje automático, la integración de enfoques neuronales y simbólicos ha surgido como una solución prometedora al problema de la explicabilidad. En este contexto, se ha demostrado que la utilización de métodos de búsqueda para la extracción de reglas en redes neuronales profundas entrenadas es eficaz. Esto implica el examen de los valores de ponderación y sesgo generados por la red, normalmente mediante el cálculo de la correlación entre los vectores de ponderación y las salidas. La hipótesis desarrollada en este artículo establece que, al incorporar la similitud de cosenos en este proceso, el espacio de búsqueda se puede reducir de manera eficiente para identificar la ruta crítica que conecta las entradas con los resultados. Además, para proporcionar una comprensión más completa e interpretable del proceso de toma de decisiones, este artículo propone la integración de la lógica de primer orden (FOL) en el proceso de extracción de reglas. Aprovechando la similitud coseno y la FOL, se diseñó e implementó un algoritmo innovador que es capaz de extraer y explicar los patrones de reglas aprendidos por una red neuronal entrenada feedforward. El algoritmo se probó en tres casos de uso y demostró su eficacia a la hora de proporcionar información sobre el comportamiento del modelo.

PALABRAS CLAVE: Aprendizaje Profundo, Explicabilidad, Eextracción de Reglas, Inteligencia Artificial, Lógica, Similitud Coseno

Introduction

Artificial deep neural networks (DNNs) have been empirically shown to have high predictive accuracy and perform well on various machine learning problems. Multilayer feedforward networks trained with the backpropagation algorithm (Rumelhart et al., 1986) are considered the most efficient method in this regard. Reasonably satisfactory answers are now available to questions such as how many examples are needed for a deep feedforward neural network to learn a concept and what is the best neural network architecture for a particular problem domain (given a fixed number of training examples), so it is now possible to train neural networks in a more efficient way.

However, it is well known that neural networks represent their knowledge in the form of numerical weights, biases, and distributed interconnections, which makes this process not understandable to the user. This represents a significant problem, given that the user is not able to understand the information output from the network or reason about the cognitive process (Lake et al., 2017; Marcus, 2018; P. W. Battaglia et al., 2018; Yann LeCun, Yoshua Bengio, 2015). Therefore, the user would have to blindly trust the response given by the network, which is clearly undesirable in several application domains, (i.e., in medical diagnosis of fatal diseases, where lives are at stake). Furthermore, if the user cannot understand or validate the discovered knowledge, they may decide to ignore it, which could lead to unwanted decision making (Garnelo et al., 2016).

The calculations performed by successive layers rarely correspond to humanly understandable reasoning steps, and the intermediate vectors of activations that are generated lack humanly understandable semantics (Nielsen et al., 2022). So, their nested and non-linear structure makes them very non-transparent, that is, it is not clear what information in the input data makes them reach their decisions. Therefore, these models are often considered black boxes (Guidotti et al. 2019).

Explainability is the ability to explain what happened when the model decided, in terms that a person understands. Its goal is to understand why a model made a particular decision and appreciate the conditions under which the model succeeds and under which it fails (Geis et al. 2019). In this sense, explainability includes both the understanding of the technical aspects of the algorithm structure and the way in which the results are presented to the user. Explainability is at the heart of responsible and open data science, across multiple industry sectors and scientific disciplines. It is a fundamental requirement for building trust with users and is the key to their safe, fair, and successful implementation in real-world applications. This is vitally important when there is legislation on algorithmic decisions made about individuals, stating that users affected by such decisions have the right to a logical explanation.

Different scientific communities have studied the problem of explaining decision models. However, each community approaches the problem from a different perspective and gives a different meaning to the explanation generated. One of the most relevant proposals for understanding the functioning of a trained neural network has focused on the extraction of symbolic rules (AmirHosseini & Hosseini, 2019; Csiszár et al., 2020; Mahdavifar & Ghorbani, 2020).

In this context, this article proposes a new technique to generate symbolic rules for explaining the behavior of an artificial neural network, emphasizing the importance of generating faithful explanations that help users to better understand the expected results of an artificial neural network.

The proposed technique consists in a combination of the application of the cosine similarity metric with first-order logic techniques. The main hypothesis of the research is that the weight vectors of the neural network can be compared using cosine similarity, to identify the inputs that best explain the outputs in each neural layer and, in this way, trace the critical path that follows the network to make decisions. So, if such a neural critical path can be identified for each layer of the network, the learned function can be represented using first-order logic, and explanation rules can be extracted. Defining a set of logical rules that capture the behavior of the network, enables reasoning about the behavior of the network using logical inference. For example, given a set of input values, it is possible to use logical inference to determine the output of the network. It is important to note that using logical rules to represent the behavior of DNNs can be challenging, as the behavior of these networks is often very complex and difficult to capture using logical rules. Furthermore, using logical inference to reason about the behavior of DNNs can be computationally expensive, especially for large networks.

The method presented in this work is in the context of Neuro-symbolic AI which is a type of artificial intelligence that integrates neural and symbolic AI architectures to address the weaknesses of each, providing a robust AI capable of reasoning, learning, and cognitive modeling.

The rest of the article is structured as follows. In Section 2, a discussion about legislation an application of the method is presented. In Section 3 the COLOSSUS algorithm is presented and described. In Subsection 3.1, the rule extraction problem is presented. In Subsection 3.2 the scope for this work is explained. In Subsection 3.3 the proposed method is introduced. In Subsection 3.4 basic notion of cosine similarity are described. Subsections 3.5, 3.6 describe different components used by the algorithm along with a detailed description of them. Section 3.7 describes the pseudocode for the algorithm proposed. In Section 4 the rule validation algorithm is described. In Section 5 use case studies along with empirical corroboration is presented. In Section 6, limitations for the study are explained. In Section 7 relevant related works are analyzed. In Section 8 the main contribution of this work is discussed. Finally, the section 9 concludes the paper.

Explainability: Legislation and Application

The utilization of data-driven methods is driving the emergence of a novel digital economy. Furthermore, the adoption of data-centric science is poised to revolutionize the way we conduct research and create solutions, as well as make decisions. Data collection, data extraction and data visualization will also be of paramount importance in scientific discovery (Montáns et al., 2019). Based on this observation, explainability addresses the critical problem that humans cannot directly understand the complex behavior of DNNs or explain their underlying decision-making process. The explainability of Machine Learning models is a fundamental requirement for building trust with users and is key to their safe, fair, and successful implementation in real-world applications.

From a legal point of view, the question of explainability goes beyond the realm of scientific interest. The adoption of the General Data Protection Regulation (GDPR) by the European Union in May 2018 (https://gdpr-info.eu/), grants any citizen the right to receive an explanation for an algorithmic decision made in relation to him. The GDPR states that individuals have the right not to be subject to a decision based solely on automated processing. Therefore, in this context, explainability is both a legal right and a responsibility that has broad social implications (Samek et al., 2017).

This paper focuses on the centrality of the human individual in the analysis of AI algorithms, specifically those employed by deep neural networks (DNNs) in decision-making processes. It is argued that any industry utilizing AI technologies which impact individuals can benefit from implementing techniques of explainability, particularly the method proposed in this article. The main aim of this research is to provide users with a comprehensible set of rules based on first-order logic that aid in the understanding of the DNN's learning process and the input variables used. Moreover, these rules can be applied to the input data, thereby generating explanations rooted in the data itself.

Although initially the method was designed to provide solutions to the capital market, ideally, the explainable method will also assist Deep Learning model development teams through data understanding, improving the hyperparameter tuning process, as well as designing the optimal DNN topology. In this sense, the training of the DNN would be more assertive, and should be able to be carried out with more discretion.

Another highly relevant area where the rule extraction method can contribute is health systems, where medical professionals could use the solution when deciding. Imaging diagnoses have a large amount of information provided by professionals associated with each image (x-ray). This tabular information associated with the image can be training input for a DNN, where the extraction method presented in this work is applied a posteriori to understand the decisions made by the DNN, and what input features it was based on.

Although the solution hasn't been tested on data sets with large amounts of features, without prejudice to the initial domain proposed, the proposed solution can be used in any solution that requires an explainability method on a DNN, trained from tabular data.

COLOSSUS: Rule Extraction with First-Order Logic for Trained Feedforward Deep Neural Networks

COLOSSUS stands for Cosmic Logic-Based Rule Extraction for Statistically Understanding Structures, reflecting the use of cosine similarity and statistical methods in this algorithm. The COLOSSUS algorithm is a rule extraction method designed with the aim of extracting logical rules from trained feedforward deep neural networks (DNN), combining first-order logic, cosine similarity and statistics obtained from the input data. DNNs are powerful and complex models that have demonstrated great success in various fields, but their lack of interpretability and transparency remains a major challenge. COLOSSUS addresses this problem by providing a systematic approach to extract understandable and interpretable rules from DNNs. First-order logic is the backbone of this algorithm, as it allows the translation of the representations learned from the DNN into logical statements. By mapping the weights and activations of neurons in the hidden layers to logical variables and predicates, the algorithm creates a rule-based representation of the DNN's decision-making process. Cosine similarity is used to identify and group similar neurons, based on their learned representations. This grouping allows the algorithm to extract more general rules, rather than individualized rules for each neuron. By considering the similarity between neurons, the algorithm can capture the important features of the input data that contribute to the final decision of the DNN. Finally, the algorithm uses statistics to refine the extracted rules. Evaluate the frequency and significance of each rule to determine its relevance and importance in the DNN decision-making process. This step ensures that only the most relevant and meaningful rules are extracted, leading to a more concise and interpretable set of rules. Overall, the COLOSSUS algorithm combines the powerful representation capabilities of first-order logic, the ability to capture general patterns using cosine similarity, and rule refinement using statistics to extract interpretable rules from DNNs. This allows for a better understanding of the decision-making process of DNNs and can help improve their performance and reliability.

Context analysis

The rule extraction task can be viewed as a search task or as a learning task (Montavon et al., 2017), where for the search approach, rules are extracted at the level of individual neurons (hidden and output). in the network, observing its weights and biases (Krishnan et al., 1999). As such, one of the main problems with the rule extraction approach is how to restrict the search space for the possible combinations of weights and biases.

In a trained neural network, the knowledge acquired in the training phase is encoded in the network architecture, the activation functions used, and the weights and biases of the neurons. In this sense, the performance of a neural network is directly related to its architecture and parameters. Therefore, the choice of an architecture for a neural network influences the learning time, predictive accuracy, noise tolerance, and generalization ability of the network (Santos et al., 2000).

In the present work, the rule extraction task consists of analyzing the weight vectors generated in each neuron, together with the bias vector for each layer of the network and extracting a set of rules from the neurons that can be explained with first-order logic (Pons et al., 2017). We consider cases where the inputs to the feedforward network are tabular and the outputs are categorical (i.e., classification problems).

The propagation method of neurons in feedforward neural networks is defined by the following canonical form:

$$Z^{[l]} = W^{\wedge[l]}\ A^{[l-1]} + b[l] \qquad \text{eq.1}$$

$$A^{[l]} = g^{[l]}\,(Z^{[l]}) \qquad \text{eq.2}$$

Where:

$W^{[l]}$	= represents the weight matrix for the neurons of layer I.
$A^{[l-1]}$	= represents the input values of layer l and where $X = A^{[0]}$.
$b^{[l]}$	= is the bias vector for layer l.
$g^{[l]}$	= is the activation function at layer l.
$A^{[l]}$	= contains the output values of layer I neurons.

The activation function for hidden layers has activations defined by:

$$ReLU = A^{[l]} = \max(0,x) \qquad \text{eq.3}$$

and for the output layer defined by:

$$\mathit{SoftMax}\text{ function } = f(z)_j = \frac{e^{zj}}{\sum_{n=1}^{m} e^{zn}} \qquad \text{eq.4.1}$$

Or

$$\mathit{Sigmoid}\text{ function } = f(z) = \frac{1}{1+e^{-z}}\ \forall\ z\ real \qquad \text{eq.4.2}$$

Scope

This work focuses on classical trained feedforward deep neural networks since the focus is on the extraction of rules from their weight matrices. Variations of this type of networks, such as recurrent neural networks, and Transformers, which, although they make use of feedforward neural networks, present different architectures since they use layers of attention, embeddings, and transformations, creating more complex hybrid architectures, are left out of this study. So, when referring to Deep Learning technologies, we refer to the concepts expressed by (Bengio et al., 2016) and (Russell & Norvig, 2010), or to an equivalent definition presented in (Domingos, 2018) where deep neural networks are presented as layers of dense neurons, which are combined with input and output layers, and some activation function in their neurons.

The proposed method

This section explains how the combination of cosine similarity and first-order logic techniques work to extract rules from a trained feedforward DNN. The weights and biases are treated with the same precision and signs as they are generated by the network, that is, no transformation is applied to them. To exemplify the present rule extraction algorithm, two three-layer feedforward networks trained using the backpropagation rule have been used (Goodfellow et al., 2016).

A neuron in each layer is considered *critical* when the calculation of the cosine similarity metric with respect to the input data with the neuron's weights exceeds a given threshold value. The procedure to extract a confirmation rule will be explained in detail.

As mentioned in Section 1, one of the most crucial issues in developing a rule extraction algorithm is how to constrain the size of the searched solution space. Each neuron within the network trains N number of weights, with N being the number of inputs for each neuron. You could think of N as the number of neurons in layer $L^{[l-1]}$. So, in a feedforward pass we will have for the $L^{[l]}$ layer, N inputs in each neuron and M outputs, with M being the number of neurons in the $L^{[l]}$ layer. Each value in the outputs of the $L^{[l-1]}$ layer neurons is presented to each $L^{[l]}$ layer neurons where they are treated with multiplications and additions and then exposed to some activation function. This is expressed in eq.1.

Then the cosine similarity will be calculated between the vector $Z^{[l]}$ of eq.1 with the weight vector formed by the weight that each neuron assigned to the feature FN. We will call this vector $T^{[l]}_{N}$.

This will allow us to understand which input features best explain the results. In this way, if after applying the cosine similarity calculation between both vectors, the metric exceeds the threshold value, we will say that the neuron associated with the feature FN is critical, and it will be included in the group of neurons in layer $L^{[l]}$ that explain the function that the network learned for a given class. In Fig. 1, each component involved in the analysis is identified.

The following section explains in detail how vector comparison helps in determining the critical path and extracting the function learned by the DNN.

Cosine Similarity

Cosine similarity is a measure of the similarity between two vectors in a space that has an inner product with which the value of the cosine of the angle between them is evaluated. This trigonometric function provides a value equal to 1 if the angle included is zero, that is, if both vectors point to the same place. Any angle existing between the vectors, the cosine would give a value less than one. If the vectors were orthogonal the cosine would cancel out, and if they pointed in the opposite direction their value would be -1. In this way, the value of this metric is between -1 and 1, that is, in the closed interval [-1,1]. Given two vectors A and B of dimension N, the cosine similarity is calculated as:

$$similarity = \cos(\emptyset) = \frac{A.B}{\|A\|.\|B\|} = \frac{\sqrt{\sum_{i=1}^{n} A_i B_i}}{\sqrt{\sum_{i=1}^{n} A_i^2}\,\sqrt{\sum_{i=1}^{n} B_i^2}} \qquad \text{eq.5}$$

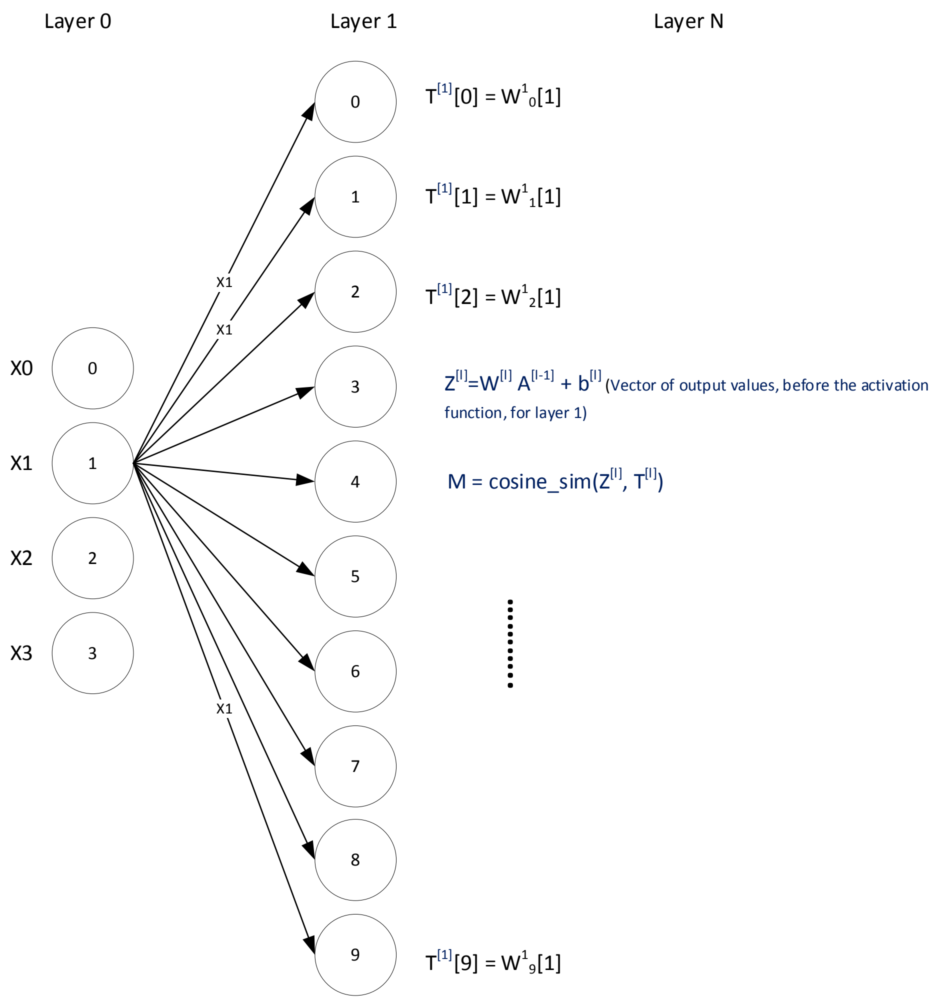
Layer 0
Layer 1
Layer N
X0
X1
X2
X3
0
1
2
3
X1
X1
X1
0
1
2
3
4
5
6
7
8
9
$T^{[1]}[0] = W^1{}_0[1]$
$T^{[1]}[1] = W^1{}_1[1]$
$T^{[1]}[2] = W^1{}_2[1]$
$Z^{[l]}=W^{[l]} A^{[l-1]} + b^{[l]}$ (Vector of output values, before the activation function, for layer 1)
$M = cosine\_sim(Z^{[l]}, T^{[l]})$
$T^{[1]}[9] = W^1{}_9[1]$

Figure 1. Calculation of cosine similarity between weight vectors

Minimum number of neurons

As explained in the following section, during the second stage of the COLOSSUS algorithm, the critical neurons in the entire network are identified, to identify a critical neural path per class (in case of classification tasks), and thus be capable of extracting the functions that the network learned and explaining them with propositional logic (Pons et al., 2017). For layer zero, where the neurons of this layer are mapped to the input data, there is a minimum set of necessary neurons (MNN), so that the algorithm that explains the rules reaches a high accuracy (acc >= 90%). This MNN set can be calculated in two ways:

1. (NFDS/2) + 1 ➔ Where NFDS= Total number of features in the set of data. For the case in which the number of neurons in the input layer is less than 10.

2. NDFS / QOL ➔ Where QOL= Number of neurons in the output layer

Then, in the case where the rule extraction method identifies fewer neurons than the MNN set, it is possible to select additional neurons as follows:

• Calculate the entropy for the features that map with the neurons of the secondary set identified in stage 2 of the algorithm.

• If the secondary set of neurons is an empty set, then the entropy is calculated over all non-critical features.

• Select the number of missing neurons taking as selection criteria the features that have the highest entropy.

• The additional neurons are added to the set of critical neurons per class for $A^{[0]}$, and are taken into account in the disjunction calculation explained in the third stage of the algorithm in the next section.

Since each feature maps to a neuron in the input layer, these neurons will be marked as critical and will be considered in the rule validation algorithm. Keep in mind that this addition of neurons is optional.

The COLOSSUS algorithm for rule extraction

The COLOSSUS rule extraction algorithm is based on the weight vector comparison strategy discussed above and works in four stages.

In the first stage, statistics are calculated for the input data for which the predictive accuracy of the network is maximized. Since COLOSSUS looks for those values that maximize the precision of the network, the critical neurons within the network will be searched with these statistical values. In addition, statistics will help in verifying the rules that explain the data. These will determine the bounds for each critical input feature. In the second stage, the cosine similarity is calculated between the vector $Z^{[l]}$ of eq.1 with the weight vector $T^{[l]}$ formed by the weight that each neuron assigned to the feature FN. with the goal of understanding which input features best explain the results. In this way, if after applying the cosine similarity calculation between both vectors the metric exceeds the threshold value, we will say that the neuron associated with the FN feature is critical, and it will be included in the group of neurons of the $L^{[l]}$ layer that will allow explain the function that the network learned for a given class. This process is repeated for each neuron in each layer.

Using the same technique, the cosine similarity is calculated between the weight vectors of the different neurons in each layer, with the goal of understanding which neurons have a relationship between their weight systems. So, if a critical neuron has a vector relationship with one or more neurons, these neurons will be said to work together, and these will be referred to as the set of secondary neurons. The MNN factor is analyzed, as explained in section 7, to complete the set of critical neurons.

In the third stage, the disjunction relations between the critical neurons of each layer are calculated for all classes, with the objective of obtaining a single truth table per layer. Secondary neurons may be added at this stage for additional validation.

To calculate the disjunction relations, the weight vectors of the critical neurons are used. If the value of the weight is positive the value will be V or 1, and if it is negative, it will be F or 0. So, if for example, for a given layer, two neurons are identified as critical, the disjunction will be calculated as $L^{[l]}_{n1} \vee L^{[l]}_{n2} = TT^{[l]}$.

Once the truth tables per layer are obtained, the implication relationship between the truth tables of each layer is calculated ($TT^{[l-1]} \rightarrow TT^{[l]} \rightarrow TT^{[l+1]} = TTdnn$). The goal of calculating the implications is to verify that the results are tautologies and ensuring that the neurons marked as critical are valid and thus guarantee that the critical neural pathway found has a valid logic-based foundation. The obtained rules are written in terms of well-formed formulas (fbf) of propositional logic (Pons et al. 2017) to obtain rules that explain the input-output relationship. These fbf explain the functions that the network learned for each class. It is important to note that, in addition to the extracted fbfs, we can also identify a regularized subnetwork that explains the class being analyzed.

In the final stage, the neurons identified as critical in the input layer $A^{[0]}$ are mapped with the statistical data obtained in the first stage, to determine the accuracy of these rules, and a distance function is calculated to solve possible overlaps between classes. As a result of this stage, first-order logic formulas are obtained (Pons et al. 2017), which explain the input data. The validation algorithm is explained in detail in the next section.

Pseudocode for the COLOSSUS algorithm

This section presents the pseudocode of the proposed algorithm. In the following sections, rule validation algorithm is analyzed and explained in detail.

```
Input

DS = aDataSet
DNN = Trained Neural Network

Output

#Step 1: Calculate statistics for DS
1.1 For the input data set calculate statistics Max, Min, Mean, Perc25, Perc50, Perc75.
1.2 Identify the statistics that maximize the predictive accuracy of the network by class.
1.3 Determine bounds for each critical input feature with point statistics.

#Step 2: Calculate cosine similarity

2.1 For each layer L in DNN
2.2     For each Neuron in the L layer
2.3         if cosine_similarity($Z^{[L]}_{n}$, $T^{[L]}_{n}$) > umbral
```

```
2.4                 critical_neurons_set.append($L^{[L]}_n$)
2.5     For each nc in critical_neurons_set:
2.6       For each ncn in non_critical_neurons_set
2.7                 If cosine_similarity(weights_nc$^{[L]}_n$, weights_ncn$^{[L]}_n$) > threshold
2.8                           seccondary_neurons_set.append(ncn$^{[L]}_n$)

# Step 3: Calculate disjunctive relations

3.1 For each layer L in DNN
3.2     #Get a single truth table per layer
3.3     TT$^{[L]}$ = calculate_disjunction (critical_neurons_set)

# Step 4: Calculate implication relationships

4.1 Taut = Calculate_implications (TTDnn)        (TT$^{[l-1]}$ → TT$^{[l]}$ → TT$^{[l+1]}$ = TTdnn)
#Write the rules obtained in terms of well-formed formulas (fbf) of propositional logic to explain
#the function learned by the DNN. Input-output relationship.

# Step 5: Calculate the MNN factor

5.1 MNN = get_MNN (critical_neurons_set (A$^{[0]}$ ) )
5.2 nAllowed = MNN - len (critical_neurons_set (A$^{[0]}$ )
5.3 do while nAllowed > 0
5.4       etpL = []
5.5       If seccondary_neurons_set != ø
5.6                 etpL = get_entropy (seccondary_neurons_set (A$^{[0]}$ )
5.7     else
5.8                 etpL = get_entropy (non_critical_neurons (A$^{[0]}$ )
5.9     critical_neurons_set.append( get_neurons (etpL,  nAllowed))
5.10    nAllowed = MNN - len (critical_neurons_set (A$^{[0]}$ ) )

# Step 6: Map critical neurons

    #Map the neurons identified as critical in critical_neurons_set for A$^{[0]}$ with the bounds of the
    #statistics obtained in the first step to determine the precision of these rules.

6.1 Calculate Max, Min for all feature F in critical_neurons_set
6.2 for each item in dataset:
6.3       for each F[item] in critical_neurons_set(A$^{[0]}$ )
6.4                 If F[item]$_n$ ⊂ [classN_max[F[item]]  .. classN_min[F[item]]
6.5                           ClassN.append( classN )

    # Resolve overlapping between classes by using the distance function.
```

```
6.6 If len(ClassN) > 1
    # Calculate distance between feature in common (fcc), with the one with the highest entropy (fme).
6.7     Class0 = (fme – MinVal_Class0(fcc)) + (MaxVal_Class0(fcc) - fme)
6.8     Class1 = (fme – MinVal_Class1(fcc)) + (MaxVal_Class1(fcc) - fme)
6.9     Class = (1 if dst1 < dst0 else 0)
6.10 Validate Precision: Ground Truth Vs. Class

#Step 7: Get First Order Logic Formulas

7.1 Write first-order logical formulas that explain the input data, out of the critical neurons in layer $A^{[0]}$ and the associated features values in the dataset.
```

Rule validation algorithm

As mentioned above, in the first stage of the rule extraction algorithm, the statistics that maximize the accuracy of the network are calculated and identified for each class. Therefore, the rule validation algorithm uses such statistics per class and per feature to use them as dimension values.

It is important to note that the rule validation algorithm operates on the union of the features identified as critical for layer $A^{[0]}$ for each class. That is, if a data set has 3 possible classes, and the extraction algorithm identifies for layer $A^{[0]}$ the following neurons per class:

1. Class 1 = N_0
2. Class 2 = N_0 ^ N_2
3. Class 3 = N_1

The validation algorithm will operate on the union of the features for all classes: if(N_0 ^ N_1 ^ N_2) ➔ClassN.

As a next step, the entire data set is read and iterated over, and it is verified that each relevant feature is within the range of values limited by the statistics obtained in the first phase. If the verification is true, we mark said instance (record within the data set) as belonging to a certain class.

If the data in the data set greatly overlap each other, it may be the case that the algorithm, for the same instance of the input data, tells us that said instance belongs to more than one class. In this case, the correct class will be determined by calculating a function that calculates the distance that the feature with the highest entropy has with respect to the centroid of the feature that the classes have in common for the overlapping classes, as follows:

• fme = It is the variable that has the highest entropy among the selected set of features.

• fcc = It is a feature that both classes have in common. (From the selected set of features. If there are no features in common or if there is more than one, choose the one with the highest entropy.)

For this example, it is assumed that the overlapping classes are classes 0 and 1.

- dst0 = (fme – MinVal_Class0(fcc)) + (MaxVal_Class0(fcc) - fme) eq. 6
- dst1 = (fme – MinVal_Class1(fcc)) + (MaxVal_Class1(fcc) - fme) eq. 7

Class = (1 if dst1 < dst0 else 0) eq. 8 ➔ The inequality can vary depending on the activation function used in the output layer.

It is important to highlight that statistics that maximize the accuracy of the network are used to identify the critical neurons and neural pathways for each class. For the rule validation algorithm and for a greater degree of generalization, the statistics for each feature are taken from its maximum and minimum values.
Finally, to measure the precision of the rule extraction algorithm, the given class *Class* is compared with the ground truth provided in the data set.

Use cases study

This section presents the results of testing the COLOSSUS algorithm on the Wine, and Breast Cancer data sets.

Example 1: The Wine dataset includes information on three classes of wines produced in Italy and 13 predictive characteristics. This data set is the result of a chemical analysis of wines grown in the same region of Italy but derived from three different cultivars. The analysis determined the quantities of 13 components found in each of the three types of wines. The dataset provides 177 examples, divided as follows: 59 examples for red wine, 71 examples for rosé wine and 48 examples for white wine.
The input features to the network for this data set are the following: *Alcohol, Malic acid, Ash, Alkalinity of Ash, Magnesium, Total Phenols, Flavonoids, Nonflavanoids Phenols, Proanthocyanins, Color Intensity, Hue, Dilution, Proline.* The expected output will be the highest accuracy in predicting the result using test data, for three classes of wine (red, rosé, and white).
For this use case, a DNN was configured with 13 neurons in the input layer, two hidden layers with 12 neurons for each layer. The *ReLU* function was used as the activation function for the hidden layers. Finally, for the output layer it was configured with 3 neurons and the SoftMax function was used as the activation function. The threshold used to determine whether a neuron is critical or not was 0.6.
The neural network trained with this data set achieved an accuracy of %92. After applying the steps identified in stages 2 and 3 of the algorithm, the following *fbf* were obtained for each class, before applying the MNN factor:

Class 0 = $N^0_{(12)}$ -> $N^1_{(10)}$ -> $N^2_{(2)}$ ^ $N^2_{(5)}$ ^ $N^2_{(8)}$ ^ $N^2_{(9)}$
Class 1 = $N^0_{(12)}$ -> $N^1_{(1)}$ ^ $N^1_{(10)}$ -> $N^2_{(5)}$ ^ $N^2_{(8)}$ ^ $N^2_{(9)}$
Class 2 = $N^0_{(12)}$ -> $N^1_{(10)}$ -> $N^2_{(5)}$ ^ $N^2_{(8)}$ ^ $N^2_{(9)}$

Table 1 presents the statistics extracted from the Wine dataset, in relation to what was stated in stage 1. Note that for classes 0 and 2, the values for the minimum have been added. Once the functions that the network has learned have been extracted, the rules are validated with the dataset data.

The feature that the algorithm identified as critical is proline *(X_{12})*, and as explained in point 2.5, it is possible to increase the number of features to be considered critical, with the aim of increasing the precision of the explanation of the data. So, for MNN 2, if we divide the number of features (13) by the number of neurons in the output layer (3), we obtain an MNN = 4. With which we can add 3 more features to the set of critics. Since for this use case the set of secondary neurons is an empty set, the entropy is calculated for all non-critical features. The analysis identifies *flavanoids* (X_6), *color_intensity* (X_9) and *malic_acid* (X_1) as critical and susceptible to being added to the set of critical neurons.

Table 1. Statistics for Wine dataset, for the classification problem

	X_1	X_6	X_9	X_{12}
	malic_acid	flavanoids	color intensity	proline
		Class-0		
Min	1,35	2,19	3,52	680
Perc50	1,77	2,98	5,4	1095
Max	4,04	3,93	8,9	1680
Dimensions	[1,35..4,04]	[2,19..3,93]	[3,52..8,9]	[680..1680]
		Class-1		
Min	0,74	0,57	1,28	278
Max	5,8	5,08	6	985
Dimensions	[0,74..5,8]	[0,57..5,08]	[1,26..6]	[278..985]
		Class-2		
Min	1,24	0,34	3,85	415
Perc50	3,27	0,685	7,55	627,5
Max	5,65	1,57	13	880
Dimensions	[1,24..5,65]	[0,34..1,57]	[3,85..13]	[415..880]

For the distance calculation, the features *proline (X_{12})* and *flavanoids (X_6)* were used since the first is the feature that was identified by the extraction algorithm and is common to all classes, and the second is the one that presents the greatest entropy, therefore the metrics for distance calculation are expressed as follows:

dstX = ($\text{proline}_{(N)}$ – MinValue_ClassX('flavanoids')) + (MaxValue_ClassX('flavanoids') - $\text{proline}_{(N)}$)
for ➔ eq. 6

dstY = ($\text{proline}_{(N)}$ – MinValue_ClassY('flavanoids')) + (MaxValue_ClassY(flavanoids') - $\text{proline}_{(N)}$)
for ➔eq. 7

After applying the calculation in eq.8, we obtain Class = (X if dstX < dstY else Y).

Finally, the value obtained in Class is compared with the ground truth for instance N. Then the rules obtained for the different classes of the Wine data set would be as follows:

Class0 = ($\forall X_1$[1,35..4,04] ∧ $\forall X_6$[2,19..3,93] ∧ $\forall X_9$ [3,52..8,9] ∧ $\forall X_{12}$[680..1680])
^ fnc_dst_class(*proline*$_{(N)}$, *flavanoids*, 0, N) == 0

$$\text{Class1} = (\forall X_1[0{,}74..5{,}8] \wedge \forall X_6[0{,}57..5{,}08] \wedge \forall X_9\ [1{,}26..6] \wedge \forall X_{12}[278..985])$$
$$\wedge\ \text{fnc\_dst\_class}(\textit{proline}_{(N)}, \textit{flavanoids}, 1, N) == 1$$

$$\text{Class2} = (\forall X_1[1{,}24..5{,}65] \wedge \forall X_6[0{,}34..1{,}57] \wedge \forall X_9\ [3{,}85..13] \wedge \forall X_{12}[415..880])$$
$$\wedge\ \text{fnc\_dst\_class}(\textit{proline}_{(N)}, \textit{flavanoids}, 2, N) == 2$$

$$\text{fnc\_dst\_class(fme, fcc, ClassX, ClassY)} = \exists X_{\boldsymbol{n}}\ (dst_x < dst_y)$$

For this data set, the rule explanation algorithm achieves an accuracy of 97%.

Example 2: The BreastCancer dataset includes information on predictions about whether a tumor is benign or malignant. Characteristics are calculated from a digitized image of a fine needle aspiration (FNAC) of a breast mass. Features describe characteristics of the cell nuclei present in the image. The input features to the network for this data set are the following: *radius_mean, texture_mean, perimeter_mean, area_mean, smoothness_mean, compactness_mean, concavity_mean, concave points_mean, symmetry_mean, fractal_dimension_mean, radius_se, texture_se, perimeter_se, area_se, smoothness_se, compactness_se, concavity_se, concave points_se, symmetry_se, fractal_dimension_se, radius_worst, texture_worst, perimeter_worst, area_worst, smoothness_worst, compactness_worst, concavity_worst, concave points_worst, symmetry_worst, fractal_dimension_worst.* The dataset contains 569 examples overall, out of which 357 correspond to benign class, and 212 correspond to malignant class.
The expected output will be the highest accuracy in predicting the outcome using test data, for two classes of tumors (benign, malignant).
For this use case, a DNN was configured with 30 neurons in the input layer, two hidden layers with 16 neurons for each layer. The *ReLU* function was used as the activation function for the hidden layers. Finally, for the output layer it was configured with 1 neuron and the *Sigmoid* function was used as the activation function. The threshold used to determine whether a neuron is critical or not was 0.95.
The neural network trained with this data set achieved an accuracy of %99.
After applying the steps identified in stages 2 and 3 of the algorithm, the following fbf were obtained.

$$\begin{aligned}\text{Class 0} = {} & N^0_{(0)} \wedge N^0_{(1)} \wedge N^0_{(3)} \wedge N^0_{(6)} \wedge N^0_{(7)} \wedge N^0_{(10)} \wedge N^0_{(12)} \wedge N^0_{(13)} \wedge N^0_{(20)} \wedge N^0_{(21)} \wedge \\ & N^0_{(22)} \wedge N^0_{(23)} \wedge N^0_{(24)} \wedge N^0_{(26)} \wedge N^0_{(27)} \wedge N^0_{(28)} \\ & \text{->} N^1_{(3)} \wedge N^1_{(5)} \wedge N^1_{(6)} \wedge N^1_{(9)} \wedge N^1_{(11)} \\ & \text{->} N^2_{(0)} \wedge N^2_{(1)} \wedge N^2_{(2)} \wedge N^2_{(3)} \wedge N^2_{(9)} \wedge N^2_{(11)} \wedge N^2_{(13)} \wedge N^2_{(15)}\end{aligned}$$

$$\begin{aligned}\text{Class 1} = {} & N^0_{(0)} \wedge N^0_{(1)} \wedge N^0_{(3)} \wedge N^0_{(6)} \wedge N^0_{(7)} \wedge N^0_{(10)} \wedge N^0_{(12)} \wedge N^0_{(13)} \wedge N^0_{(20)} \wedge N^0_{(21)} \wedge \\ & N^0_{(22)} \wedge N^0_{(23)} \wedge N^0_{(24)} \wedge N^0_{(26)} \wedge N^0_{(27)} \wedge N^0_{(28)} \\ & \text{->} N^1_{(3)} \wedge N^1_{(5)} \wedge N^1_{(6)} \wedge N^1_{(9)} \wedge N^1_{(11)} \\ & \text{->} N^2_{(0)} \wedge N^2_{(1)} \wedge N^2_{(2)} \wedge N^2_{(3)} \wedge N^2_{(9)} \wedge N^2_{(11)} \wedge N^2_{(13)} \wedge N^2_{(15)}\end{aligned}$$

As we can see, both fbfs are identical. This is so, since the treatment approach for this data set can be seen as a classification or prediction problem. In this sense, and for prediction problems, the algorithm will deliver a normalized network and a unique set of rules for both classes.

Table 2. Statistics for the BreastCancer dataset, for the classification problem.

Stat	X_0	X_1	X_2	X_3	X_6	X_7	X_{10}	X_{12}	X_{13}	X_{20}	X_{21}	X_{22}	X_{23}	X_{24}	X_{26}	X_{27}	X_{28}
Clase 0																	
Min	6.98	9.71	43.79	143.5	0	0	0.115	0.758	6.802	7.93	12.02	50.41	185.2	0.07117	0	0	0.1566
P75	13.37	19.76	86.1	551.1	0.05999	0.03251	0.3416	2.388	25.03	14.8	26.51	96.59	670	0.1376	0.2216	0.0974	0.2983
Max	17.85	33.81	114.2	992.1	0.4108	0.08534	0.811	5.118	77.11	19.82	41.78	127.1	1210	0.2006	1.252	0.175	0.4228
Clase 1																	
Min	10.95	10.38	71.9	361.6	0.02398	0.0231	0.1938	1.334	13.99	12.84	16.67	85.1	508.1	0.08822	0.0239	0.0289	0.1565
P50	17.32	21.46	114.2	932	0.1513	0.08628	0.5472	3.6795	54.455	20.59	28.945	138	1303	0.14345	0.4049	0.182	0.3103
Max	28.11	39.28	188.5	2501	0.4268	0.2012	2.873	21.98	542.2	36.04	49.54	251.2	4254	0.2226	1.17	0.291	0.6638

Table 3. References and boundaries by Feature for the BreastCancer Dataset, for the Classification.

#	Features	Dimensions Class 0	Dimensions Class 1
X0	radius mean	[6.98..17.85]	[10.95..28.11]
X1	texture mean	[9.71..33,81]	[10,38..39.28]
X2	perimeter mean	[43.79..114,2]	[71,9..188.5]
X3	area mean	[143.5..992,1]	[361,6..2501]
X6	concavity mean	[0..0.4108]	[0.02398..0.4268]
X7	concave points mean	[0..0.08534]	[0,0231..0.2012]
X10	radius se	[0.115..0.811]	[0.1938..2.873]
X12	perimeter se	[0.758..5,118]	[1,334..5,118]
X13	area se	[6.802..77,11]	[13,99..542.2]
X20	radius worst	[7.93..19,82]	[12,84..36.04]
X21	texture worst	[12.02..41,78]	[16,67..49.54]
X22	perimeter worst	[50.41.. 127,1]	[85,1..251.2]
X23	area worst	[185.2..1210]	[508,1..4254]
X24	smoothness worst	[0.07117.. 0.2006]	[0,08822.. 0.2226]
X26	concavity worst	[0.. 1,252]	[0,0239..1.17]
X27	concave points worst	[0.. 0.0175]	[0,0289..0.291]
X28	symmetry worst	[0.1566.. 0.4228]	[0,1565.. 0.6638]

Table 2 presents the statistics extracted from the BreastCancer dataset, in relation to what was stated in stage 3. Note that for classes 0 the values for the maximum have been added, while for class 1 the values have been added for the minimum. This is as a previous step to calculating the distance of the common features, as an additional method of checking whether extending the limits of the dimensions between the minimum and maximum values of each feature helps to resolve the overlap between classes.

Once the functions that the network learned have been extracted, and the statistics for each class have also been calculated, the rules are validated with the dataset data in its entirety. For this case, the relevant features for the 2 classes are the same and these are presented in Table 2 and Table 3.

For this case, the features that the algorithm identified as critical are sufficient, therefore, it is not necessary to increase the number of features to be considered critical using neurons from the secondary set.

For the distance calculation, the features area worst *(X_{23})* and radius mean *(X_0)* were used since these are the features identified by the extraction algorithm, and both are features in common between the classes. Of both, the one with the greatest entropy is the area worst (X_{23}), therefore, the metrics for calculating distance are expressed as follows:

dstX = (*area_worst*$_{(N)}$ – MinValue_ClassX('*radius_mean*')) +
(MaxValue_ClassX('*radius_mean*') – *area_worst* $_{(N)}$) for ➔ eq. 6

dsty = (*area_worst*$_{(N)}$ – MinValue_ClassY('*radius_mean*')) +
(MaxValue_ClassY('*radius_mean*') – *area_worst* $_{(N)}$) for ➔ eq. 7

After applying the calculation in eq. 8, we get Class = (X if dstX < dstY else Y).

Finally, the value obtained in Class is compared with the ground truth for instance N. Then the rule obtained for the different classes of the Breast Cancer data set would be as follows:

Class0 = ($\forall X_0$[6,98..13.37] $\wedge$ $\forall X_1$[9,71..19.76] $\wedge$ $\forall X_2$ [43,79..86,1] $\wedge$ $\forall X_3$[143,5..992,1] $\wedge$ $\forall X_6$[0..0,4108] $\wedge$ $\forall X_7$[0..0,08534] $\wedge$ $\forall X_{10}$[0,115..0,811] $\wedge$ $\forall X_{12}$[0,758..5,118] $\wedge$ $\forall X_{13}$[6,802..77,11] $\wedge$ $\forall X_{20}$[7,93..19,82] $\wedge$ $\forall X_{21}$[12,02..41,78] $\wedge$ $\forall X_{22}$[50,41.. 127,1] $\wedge$ $\forall X_{23}$[185,2..1210] $\wedge$ $\forall X_{24}$[0,07117.. 0.2006] $\wedge$ $\forall X_{26}$[0.. 1,252] $\wedge$ $\forall X_{27}$[0.. 0,0175] $\wedge$ $\forall X_{28}$[0,1566.. 0,4228])
^ fnc_dst_class(*area_worst*$_{(N)}$, *radius_mean*, 0, N) == 0

Class1 = ($\forall X_0$[10,95..28,11] $\wedge$ $\forall X_1$[10,38..39,28] $\wedge$ $\forall X_2$[71,9..188,5] $\wedge$ $\forall X_3$[361,6..2501] $\wedge$ $\forall X_6$[0,02398..0,4268] $\wedge$ $\forall X_7$[0,0231..0,2012] $\wedge$ $\forall X_{10}$[0,1938..2,873] $\wedge$ $\forall X_{12}$[1,334..5,118] $\wedge$ $\forall X_{13}$[13,99..542,2] $\wedge$ $\forall X_{20}$[12,84..36,04] $\wedge$ $\forall X_{21}$ [16,67..49,54] $\wedge$ $\forall X_{22}$ [85,1..251,2] $\wedge$ $\forall X_{23}$[508,1..4254] $\wedge$ $\forall X_{24}$ [0,08822.. 0,2226] $\wedge$ $\forall X_{26}$[0,0239..1.17] $\wedge$ $\forall X_{27}$ [0.0289..0.291] $\wedge$ $\forall X_{28}$ [0,1565.. 0.6638])
^ fnc_dst_class(*area_worst*$_{(N)}$, *radius_mean*, 1, N) == 1

fnc_dst_class(fme, fcc, ClassX, ClassY) = $\exists$X$\boldsymbol{n}$ (dstx < dsty). For this data set, the rule explanation algorithm achieves an accuracy of 95%.

Limitations of the Study and Future Work

The empirical results reported herein should be considered in the light of some limitations.

For the case study introduced in section 5, well-known data sets widely used by the scientific community were applied. This decision was made for the purpose of reproducibility, availability, and understandability of the results. In this way, it was possible to provide an explanation of the method, its results, and the connection of the results with the input data. While using these well-known datasets to evaluate the proposal is a pertinent and useful strategy, it can also be seen as

a limitation, as larger, more sophisticated datasets that require more complex DNNs could lead to unforeseen challenges.
To overcome this limitation, a further evaluation of the algorithm with real data is under development. In particular, the algorithm was tested on a dataset provided by a health institution, even though the algorithm performs as expected, the results are not authorized to be published yet.

Given their nature of not being regularly disseminated through publication, but through unconventional channels that are difficult to access, articles classified as gray bibliography may have been outside the scope of the review as they were not found by search engines.

Presence of segmented publication, which is a distinct form of redundant publication that is generally characterized by similarity of hypotheses, methodology or results, but not by similarity of text. These aspects of the publications are not objectively detected by software applications and therefore represent a serious threat to the review analysis.

Future lines of work are summarized in the following points outlined below.

Investigate the application of the proposed method in more complex and larger scale neural networks to evaluate their scalability and performance.

Conduct the study on bigger datasets from the industry that demand the development of complex and sophisticated DNN topologies.

Explore the integration of additional logical frameworks, such as fuzzy logic or probabilistic logic, to improve the interpretability and explainability of the extracted rule patterns.

Conduct empirical studies to evaluate the effectiveness of the method in different tasks and domains, such as capital market, banking, and health, to evaluate its generalizability and accuracy and robustness. Additionally, to benchmarking with other models to evaluate accuracy, readability and interpretability of the extracted rules.

Examine the impact of different network architectures and training techniques on the interpretability of extracted rule patterns, to gain insight into the relationship between network structure and explainability.

Investigate the potential of the proposed method in combination with other explainability techniques, such as attention mechanisms or saliency maps, to provide a more complete and detailed understanding of the decision-making process in neural networks.

Related Work

Today, non-symbolic AI is the most attractive area of AI which is related to machine learning (Alpaydin, 2020). Several researchers have proposed algorithms to extract rules from a trained feedforward neural network, where the explainability problem was addressed form different perspectives, as follows:

To address the lack of explainability (AmirHosseini & Hosseini, 2019)fuzzy systems have high level of interpretability because of using linguistic terms for knowledge representation through the reasoning process. The evolutionary algorithms can be applied for optimization of the systems. This article takes advantages of differential evolutionary algorithm to search the problem space more intelligently using the knowledge of distance vector of the candidate solutions. For this, a hybrid fuzzy-DE model has been proposed for the problem of classification of the Haptic metastasis tumors through information obtained from the features measured in the CT scan images by radiologists. The hybrid fuzzy-DE model was evaluated using a real liver cancer dataset obtained from the Noor medical imaging center in Tehran. The results of the hybrid proposed models were compared with the diagnosis of the specialists, the results reveal that the proposed fuzzy-DE model has high capability for diagnosis of the hepatic metastasis tumors with an accuracy of 99.24% with 95% confidence interval (98.32 100 propose improving the trade-offs between the accuracy and interpretability of the fuzzy inference system (FIS) by adjusting the model parameters using an evolutionary algorithm that explores the search space more smarter than other proposed algorithms.
In (Mahdavifar & Ghorbani, 2020) the authors propose a deep neural network expert system (DenNES) that extracts refined rules from a trained deep neural network (DNN) architecture to replace the knowledge base of an expert system.
For their part (Cocarascu et al., 2018), they present a methodology called ANNA (Artificial Neural Networks & Abstract Augmentation) that combines artificial neural networks and abstract argumentation for prediction. ANNA provides predictions based on logical arguments and rules that offer equivalent predictions.
In the work developed in (Csiszár et al., 2020), the authors propose a coherent framework to model human thinking through the use of multi-criteria decision making (MCDM) tools and fuzzy logic.
The objective of the project in (Schmid & Finzel, 2020), is to combine black box deep learning approaches, with interpretable machine learning for the classification of different types of medical images, to combine the predictive accuracy of deep learning and the transparency and understandability of interpretable models.

In the study presented in (Dombi & Csiszár, 2021)we demonstrate how combining neural networks with continuous logic and multi-criteria decision-making tools can reduce the black-box nature of neural models. We show that nilpotent logical systems studied in the previous chapters offer an appropriate mathematical framework for the hybridization of continuous nilpotent logic and neural models, and help improve the interpretability and safety of machine learning. In this approach, perceptrons model soft inequalities; namely membership functions and continuous logical operators. We design the network architecture before training, using continuous logical operators and multi-criteria decision tools with given weights working in the hidden layers. Designing the structure appropriately leads to a drastic reduction in the number of parameters to be learned. Our theoretical basis offers a straightforward choice of activation functions (the cutting function or its differentiable approximation, the squashing function, the authors demonstrate that nilpotent logic systems offer an appropriate mathematical framework for a hybridization of continuous nilpotent logic and neural models, which helps to improve the interpretability and security of machine learning.

In (Aghaeipoor et al., 2023), the authors propose rule-based fuzzy explanatory systems for deep neural networks, which can be used for local and global explainability purposes. The algorithm learns a compact but precise set of fuzzy rules based on the importance of features (i.e., attribution values) extracted from the trained networks.

(Ciravegna et al., 2023)that consists in their lack of capability in providing human-understandable motivations of their decisions. In situations in which the machine is expected to support the decision of human experts, providing a comprehensible explanation is a feature of crucial importance. The language used to communicate the explanations must be formal enough to be implementable in a machine and friendly enough to be understandable by a wide audience. In this paper, we propose a general approach to Explainable Artificial Intelligence in the case of neural architectures, showing how a mindful design of the networks leads to a family of interpretable deep learning models called Logic Explained Networks (LENs presents the fundamental methods used to implement Logic Explained Networks (LEN). The authors describe the procedure used to extract logical rules from LENs for an individual observation or a group of samples.

In (Garcez et al., 2019), neural symbolic computing for integrated machine learning and reasoning is discussed. It highlights the importance of integrating neural learning with robust symbolism-based reasoning methods to develop explainable and responsible systems and tools based on AI and machine learning.

In (Burkhardt et al., 2021) a method is proposed to extract logical rules from binary neural networks using first-order convolutional rules. The method is based on rule extraction from binary neural networks with stochastic local search and can be used to extract interpretable rules from binary neural networks, validate models, and high-dimensional input data such as images.

In (Barbiero et al., 2022)most of these approaches focus on the identification of the most relevant concepts but do not provide concise, formal explanations of how such concepts are leveraged by the classifier to make predictions. In this paper, we propose a novel end-to-end differentiable approach enabling the extraction of logic explanations from neural networks using the formalism of First-Order Logic. The method relies on an entropy-based criterion which automatically identifies the most relevant concepts. We consider four different case studies to demonstrate that: (i, a novel end-to-end differentiable approach is proposed that allows logical explanations to be extracted from neural networks using the formalism of first-order logic. The method is based on an entropy-based criterion that automatically identifies the most relevant concepts. The article considers four different case studies to demonstrate that this entropy-based criterion allows concise logical explanations to be extracted in safety-critical domains, from clinical data to computer vision.

In the work developed in (Tran, 2017), a method is proposed to integrate symbolic knowledge in unsupervised neural networks. The method is based on the theoretical finding that any propositional formula can be represented in restricted Boltzmann machines (RBM).

In (Dombi & Csiszár, 2021)we demonstrate how combining neural networks with continuous logic and multi-criteria decision-making tools can reduce the black-box nature of neural models. We show that nilpotent logical systems studied in the previous chapters offer an appropriate mathematical framework for the hybridization of continuous nilpotent logic and neural models, and help improve the interpretability and safety of machine learning. In this approach, perceptrons model soft inequalities; namely membership functions and continuous logical operators. We design the network architecture before training, using continuous logical operators and multi-criteria decision tools with given weights working in the hidden layers. Designing the structure appropriately leads to a drastic reduction in the number of parameters to be learned. Our theoretical basis offers a straightforward choice of activation functions (the cutting function or its differentiable approximation, the squashing function, the authors propose a hybrid approach that combines neural networks with continuous logic and multi-criteria decision-making tools to improve the interpretability and security of machine learning. The approach uses continuous nilpotent logic and continuous logical operators to model soft inequalities and reduces the number of parameters that must be learned. Practical implications include improving the interpretability and security of machine learning in various domains, such as healthcare, finance, and engineering.

In the work presented in (Wang & Pan, 2020), the authors propose a framework that integrates logical knowledge in the form of first-order logic into a deep learning system for information extraction.

(Giunchiglia et al., 2022), discusses several methods that use constraints and logical reasoning in deep learning models to achieve better performance, learn from less data, and ensure compliance with basic knowledge for safety-critical applications.

In (Dai & Muggleton, 2021)it is challenging to design an appropriate endto-end learning pipeline. Neuro-Symbolic Learning, divide the process into sub-symbolic perception and symbolic reasoning, trying to utilise datadriven machine learning and knowledge-driven reasoning simultaneously. However, they suffer from the exponential computational complexity within the interface between these two components, where the sub-symbolic learning model lacks direct supervision, and the symbolic model lacks accurate input facts. Hence, most of them assume the existence of a strong symbolic knowledge base and only learn the perception model while avoiding a crucial problem: where does the knowledge come from? In this paper, we present Abductive Meta-Interpretive Learning (M etaAbd, the authors propose an innovative approach called abductive meta interpretive learning (Meta Abd) that combines abduction and induction to learn neural networks and jointly induce first-order logical theories from raw data.

(Zarlenga et al., 2021) present ECLAIRE, a novel polynomial-time rule extraction algorithm capable of scaling to both large neural network architectures and large training data sets.

In (P. A. Negro & Pons, 2023), the authors propose an innovative method to extract the rule pattern learned by a feed-forward trained neural network, analyze its properties and explain these patterns through the use of first-order logic (FOL), by properly ordering the input weights of a neuron, narrowing down the search space.

In (De et al., 2020) the authors introduce a hybrid approach that combines clustering and decision trees to explain the predictions made by deep neural networks. This method allows for the visualization of information flow in the network, providing human-interpretable explanations for individual predictions. The effectiveness of this approach is demonstrated in the context of credit card default prediction, showing its ability to generate precise and localized reason codes.

In (Angelov & Soares, 2019)we propose an elegant solution that is directly addressing the bottlenecks of the traditional deep learning approaches and offers a clearly explainable internal

architecture that can outperform the existing methods, requires very little computational resources (no need for GPUs authors proposes a new deep learning architecture, called xDNN, which overcomes the limitations of traditional methods by offering a transparent internal architecture. It utilizes prototypes derived from training data to efficiently combine reasoning and learning. The model is non-iterative and non-parametric, resulting in improved efficiency in terms of time and computational resources. Its transparency, with prototype-based IF...THEN rules, makes it suitable for high-stakes applications in fields such as autonomous vehicles and healthcare.
(Samek et al., 2021) offers a comprehensive overview of explainable AI techniques for deep neural networks, with a focus on post hoc explanations. It discusses various methods and applications for interpreting the predictions made by deep learning models, highlighting the importance of explanations in this context. The text covers popular approaches for explaining deep neural networks, aiding researchers, and practitioners in choosing the most suitable technique for their specific use case. These explanations can enhance the understanding of model behavior, identify biases, and provide insights into decision-making processes, ultimately facilitating the development and deployment of more transparent and trustworthy deep learning models. This is particularly relevant in domains where interpretability is crucial, such as healthcare, finance, and autonomous systems.

(Amarasinghe et al., 2018) proposes a new deep neural network architecture for anomaly detection that combines convolutional and recurrent neural networks. It also introduces a new dataset for evaluating anomaly detection methods in the context of cybersecurity and provides insights into the interpretability of deep learning models for this task. The proposed model offers a new approach that combines deep learning techniques with interpretability to improve performance and understanding of anomaly detection systems. The practical implications of this research include enhanced anomaly detection performance, interpretability in anomaly detection, improved cybersecurity measures, and guidance for future research in developing transparent and reliable anomaly detection systems.

(Shahroudnejad, 2021) discusses key contributions in the field of deep neural networks (DNNs) understanding, visualizations, and explanations. These contributions include an overview of methods for visualization and understanding, the concept of explainability in DNNs, structural and behavioral analysis, and the development of interpretable models. These practical implications have implications in enhancing model interpretability, improving trust in AI systems, developing explainable AI, and applying DNNs in healthcare and computer vision. Additionally, ethical considerations are highlighted in the context of understanding the rationale behind AI decisions.

The interested reader may consider reading an exhaustive study conducted in the systematic literature review published as a prelude to this work in (P. Negro & Pons, 2022).

Main Contributions

The main contributions made in this work are:

- A new algorithm is proposed to extract the learned rule pattern from a trained feedforward neural network and analyze its properties.

- Propositional logic and first order logic (FOL) are both used to explain the patterns learned by the neural network.
- A white box method that analyzes the weight matrices of DNNs to extract rules, with the cosine similarity metric is presented.
- A distance function is implemented to resolve overlap between classes.
- The method does not require any special learning rules during rule extraction, making it easy to use.
- The method works efficiently for both continuous and enumerated attributes.
- The algorithm is very efficient in identifying relevant neurons from the DNN and does not require retraining after selection of critical neurons.
- The extracted rules can be used to explain the decision-making process of the neural network, through first-order logic formulas, which can be useful in various applications, such as medical diagnosis, fraud detection and the credit rating.
- The extracted rules are mapped directly to the data set, and the network's decisions can be explained with data.
- Indirectly, the rule extraction algorithm proposes a regularization method for the DNN, by identifying the critical neural paths, thus composing smaller networks.

Conclusions

This article describes the COLOSSUS algorithm that generates symbolic rules to explain the behavior of an artificial neural network. The algorithm begins with a process of calculating statistics on the data, and subsequent extraction of weights from the trained neural network, where the cosine similarity between the vectors is calculated, which stablishes each input variable for each neuron in relation to each other, and in relation to the bias vector, and the vector formed by the activation values in each layer. Furthermore, the same procedure is performed for the weight vectors of each neuron, which results in an interneuron comparison of weight vectors for neurons in the same layer, with the goal of understanding the collaboration between the neurons of a given layer.

The resulting rules are presented in well-formed formulas of propositional logic, providing insight into the correlation between input variables and the network's output. This method identifies critical neural paths by analyzing the data flow from input to output and delivers regularized networks for each class. The extracted rules are based on the statistics obtained from the input data and are chosen for their high precision in explaining the network's results. The alignment between these metrics serves as a foundation for validating and verifying the extracted rules.

The neurons identified by the algorithm in the input layer are mapped to the features of the dataset, where the rule validation algorithm operates on these features and the statistics previously collected, and a distance function is also utilized to resolve overlaps.

One of the major strengths of this technique is its simplicity and efficiency, as it does not require prior learning rules for extracting rules. It is also adaptable to various types of attributes. Additionally, the algorithm produces complete and valid rules, as well as identifying important neurons and critical neural pathways. Because of the use of first-order logic, these rules can be easily interpreted, making them highly valuable for a range of applications.

References

» Aghaeipoor, F., Sabokrou, M., & Fernández, A. (2023). Fuzzy Rule-Based Explainer Systems for Deep Neural Networks: From Local Explainability to Global Understanding. IEEE Transactions on Fuzzy Systems, 1–12. https://doi.org/10.1109/TFUZZ.2023.3243935

» Alpaydin, E. (2020). Introduction to Machine Learning. MIT Press.

» Amarasinghe, K., Kenney, K., & Manic, M. (2018). Toward Explainable Deep Neural Network Based Anomaly Detection. 2018 11th International Conference on Human System Interaction (HSI), 311–317. https://doi.org/10.1109/HSI.2018.8430788

» AmirHosseini, B., & Hosseini, R. (2019). An improved fuzzy-differential evolution approach applied to classification of tumors in liver CT scan images. Medical & Biological Engineering & Computing, 57(10), Article 10. https://doi.org/10.1007/s11517-019-02009-7

» Angelov, P., & Soares, E. (2019). Towards Explainable Deep Neural Networks (xDNN) (arXiv:1912.02523). arXiv. http://arxiv.org/abs/1912.02523

» Barbiero, P., Ciravegna, G., Giannini, F., Lió, P., Gori, M., & Melacci, S. (2022). Entropy-Based Logic Explanations of Neural Networks. Proceedings of the AAAI Conference on Artificial Intelligence, 36(6), 6046–6054. https://doi.org/10.1609/aaai.v36i6.20551

» Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., … Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks. arXiv:1806.01261 [Cs, Stat]. http://arxiv.org/abs/1806.01261

» Bengio, Y., Goodfellow, I., & Courville, A. (2016, November 29). Deep Learning. https://www.deeplearningbook.org/

» Burkhardt, S., Brugger, J., Wagner, N., Ahmadi, Z., Kersting, K., & Kramer, S. (2021). Rule Extraction From Binary Neural Networks With Convolutional Rules for Model Validation. Frontiers in Artificial Intelligence, 4, 642263. https://doi.org/10.3389/frai.2021.642263

» Ciravegna, G., Barbiero, P., Giannini, F., Gori, M., Lió, P., Maggini, M., & Melacci, S. (2023). Logic Explained Networks. Artificial Intelligence, 314, 103822. https://doi.org/10.1016/j.artint.2022.103822

» Cocarascu, O., Cyras, K., & Toni, F. (2018). Explanatory predictions with artificial neural networks and argumentation.

» Csiszár, O., Csiszár, G., & Dombi, J. (2020). How to implement MCDM tools and continuous logic into neural computation?: Towards better interpretability of neural networks. Knowledge-Based Systems, 210, 106530. https://doi.org/10.1016/j.knosys.2020.106530

» Dai, W.-Z., & Muggleton, S. H. (2021). Abductive Knowledge Induction From Raw Data (arXiv:2010.03514). arXiv. http://arxiv.org/abs/2010.03514

» De, T., Giri, P., Mevawala, A., Nemani, R., & Deo, A. (2020). Explainable AI: A Hybrid Approach to Generate Human-Interpretable Explanation for Deep Learning Prediction. Procedia Computer Science, 168, 40–48. https://doi.org/10.1016/j.procs.2020.02.255

» Dombi, J., & Csiszár, O. (2021). Interpretable Neural Networks Based on Continuous-Valued Logic and Multi-criteria Decision Operators. In J. Dombi & O. Csiszár (Eds.), Explainable Neural Networks Based on Fuzzy Logic and Multi-criteria Decision Tools (pp. 147–169). Springer International Publishing. https://doi.org/10.1007/978-3-030-72280-7_9

» Domingos, P. (2018). How the Quest for the Ultimate Learning Machine will remake our World. In The Master Algorithm How.

» Garcez, A. d'Avila, Gori, M., Lamb, L. C., Serafini, L., Spranger, M., & Tran, S. N. (2019). Neural-Symbolic Computing: An Effective Methodology for Principled Integration of Machine Learning and Reasoning (arXiv:1905.06088). arXiv. http://arxiv.org/abs/1905.06088

» Garnelo, M., Arulkumaran, K., & Shanahan, M. (2016). Towards Deep Symbolic Reinforcement Learning. arXiv:1609.05518 [Cs]. http://arxiv.org/abs/1609.05518

» Giunchiglia, E., Stoian, M. C., & Lukasiewicz, T. (2022). Deep Learning with Logical Constraints (arXiv:2205.00523). arXiv. http://arxiv.org/abs/2205.00523

» Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. The MIT Press.

» Krishnan, R., Sivakumar, G., & Bhattacharya, P. (1999). Extracting decision trees from trained neural networks. 12.

» Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. Behavioral and Brain Sciences. https://doi.org/10.1017/S0140525X16001837

» Mahdavifar, S., & Ghorbani, A. A. (2020). DeNNeS: Deep embedded neural network expert system for detecting cyber attacks. Neural Computing and Applications, 32(18), Article 18. https://doi.org/10.1007/s00521-020-04830-w

» Marcus, G. (2018). Deep Learning: A Critical Appraisal. arXiv:1801.00631 [Cs, Stat]. http://arxiv.org/abs/1801.00631

» Montavon, G., Lapuschkin, S., Binder, A., Samek, W., & Müller, K.-R. (2017). Explaining nonlinear classification decisions with deep Taylor decomposition. Pattern Recognition, 65, 211–222. https://doi.org/10.1016/j.patcog.2016.11.008

» Negro, P. A., & Pons, C. (2023). Extracción de reglas en redes neuronales feedforward entrenadas con lógica de primer orden. Memorias de las JAIIO, 9(2), Article 2.

» Negro, P., & Pons, C. (2022). Artificial Intelligence techniques based on the integration of symbolic logic and deep neural networks: A systematic review of the literature. Inteligencia Artificial, 25(69), 13–41. https://doi.org/10.4114/intartif.vol25iss69pp13-41

» Nielsen, I. E., Dera, D., Rasool, G., Bouaynaya, N., & Ramachandran, R. P. (2022). Robust Explainability: A Tutorial on Gradient-Based Attribution Methods for Deep Neural Networks. IEEE Signal Processing Magazine, 39(4), 73–84. https://doi.org/10.1109/MSP.2022.3142719

» Pons, C., Rosenfeld, R., & Smith, C. P. (2017). Lógica para Informática. Editorial de la Universidad Nacional de La Plata (EDULP). https://doi.org/10.35537/10915/61426

» Rumelhart, D. E., Hintont, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors.

» Russell, S., & Norvig, P. (2010). Artificial Intelligence A Modern Approach Third Edition. In Pearson. https://doi.org/10.1017/S0269888900007724

» Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., & Muller, K.-R. (2021). Explaining Deep Neural Networks and Beyond: A Review of Methods and Applications. Proceedings of the IEEE, 109(3), 247–278. https://doi.org/10.1109/JPROC.2021.3060483

» Samek, W., Wiegand, T., & Müller, K.-R. (2017). Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models (arXiv:1708.08296). arXiv. http://arxiv.org/abs/1708.08296

» Santos, R. T., Nievola, J. C., & Freitas, A. A. (2000). Extracting comprehensible rules from neural networks via genetic algorithms. 2000 IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks. Proceedings of the First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks (Cat. No.00EX448), 130–139. https://doi.org/10.1109/ECNN.2000.886228

» Schmid, U., & Finzel, B. (2020). Mutual Explanations for Cooperative Decision Making in Medicine. KI - Kunstliche Intelligenz, 34(2), Article 2. https://doi.org/10.1007/s13218-020-00633-2

» Shahroudnejad, A. (2021). A Survey on Understanding, Visualizations, and Explanation of Deep Neural

Networks (arXiv:2102.01792). arXiv. http://arxiv.org/abs/2102.01792

» Tran, S. N. (2017). Unsupervised Neural-Symbolic Integration (arXiv:1706.01991). arXiv. http://arxiv.org/abs/1706.01991

» Wang, W., & Pan, S. J. (2020). Integrating deep learning with logic fusion for information extraction. Proceedings of the AAAI Conference on Artificial Intelligence, 34(05), 9225–9232.

» Yann LeCun, Yoshua Bengio, G. H. (2015). Deep learning (2015), Y. LeCun, Y. Bengio and G. Hinton. Nature.

» Zarlenga, M. E., Shams, Z., & Jamnik, M. (2021). Efficient Decompositional Rule Extraction for Deep Neural Networks (arXiv:2111.12628). arXiv. https://doi.org/10.48550/arXiv.2111.12628