



# Comprehensible and transparent rule extraction using neural network

Saroj Kr. Biswas<sup>1</sup> · Arijit Bhattacharya<sup>2</sup>  · Abhinaba Duttachoudhury<sup>1</sup> · Manomita Chakraborty<sup>3</sup> · Akhil Kumar Das<sup>2</sup>

Received: 5 June 2023 / Revised: 25 October 2023 / Accepted: 9 January 2024 /

Published online: 6 February 2024

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

## Abstract

In data mining and machine learning communities, Neural Network (NN) is a popular classification method. On extremely unbalanced and complicated datasets, NN may achieve excellent classification accuracy. However, one disadvantage of NN is its inability to explain its reasoning process, which restricts its use in numerous sectors that need clear conclusions as well as high accuracy. To address this issue, rule-extraction mechanisms exist that extract intelligible classification-rules from NN and turn them into a white box. Attribute or network pruning, dealing with diverse attribute types, rule pruning, and dealing with class overlapping difficulties are all significant components or portions of many existing rule extraction methods, and present strategies to deal with these aspects are insufficiently successful. As a result, this study offers a rule extraction approach named “Comprehensible and Transparent Rule Extraction Using Neural Network”-CTRENN to address the aforementioned shortcomings and transform NN into a white box with high accuracy and better explain-ability. The suggested CTRENN is an expansion of the state-of-art Rule Extraction from Neural Network Using Classified and Misclassified Data technique (RxNCM). The CTRENN augments the RxNCM with a floating sequential search for feature and rule selection to improve feature and rule selection. CTRENN also distinguishes between continuous and discrete properties to improve the readability of the produced rules. Unlike RxNCM, the CTRENN employs a probabilistic technique to deal with the overlapping of attribute data ranges in various classes. Experiments are carried out using six real life datasets obtained from the UCI repository in order to illustrate the efficacy of the proposed CTRENN algorithm in comparison to the current methods.

**Keywords** Neural Networks · Rule extraction · Classification · RxREN · RxNCM

## 1 Introduction

In our digital era, massive volumes of data are acquired in various forms from many sources on a daily basis. This gathered data contains a wealth of important information that is challenging to extract appropriately. In response, data mining techniques [1] have

developed supercomputing capabilities that can mine unseen patterns and information and apply them to various decision-making situations. Tasks related to data mining generally include regression, classification [2], clustering [3], association analysis [4], and so on. Among these tasks' classification is the most common and popular [5]. Bayesian Classification [6], Decision Trees [7, 30, 31], Ensembles [8], Neural Networks [9], SVM [31], and other classification approaches are widely used and dominating. Among these approaches, the utilization of Neural Networks in Data Mining activity has increased in recent years due to their unparalleled capabilities of classifying data with mixed-mode properties, obtaining higher precision or accuracy, and keeping low computing complexity [10, 11]. The disadvantage of Neural Networks in decision-making is their black-box character, which is incapable of explaining the decision-making process in a comprehensible manner [12]. Because the black-box nature [13] of Neural Networks makes them incomprehensible in thinking and decision making, they suffer in sectors where an explanation for the conclusion is required [31, 32]. Medical diagnosis, financial decision-making, infrastructure management, and other professions sometimes demand a clear explanation of rationale and decision process. In medical diagnostics, for example, a clear explanation of the etiology of a disease is essential to raise awareness among the general public and to take preventative steps to prevent the sickness from spreading. As a result, several methods have been developed to extract intelligible rules from neural networks in order to turn the black-box nature of neural networks into white boxes [14].

Considering the approaches used to extract the rules, neural network rule extraction strategies can be classified as decompositional, pedagogical, or eclectic [15]. Analyzing the weights between units and activation functions is part of decompositional approaches. The association between the inputs and outcomes is examined in pedagogical approaches to derive rules. Eclectic methods include decompositional and pedagogical strategies together [16]. Because of lower processing in terms of computational requirement, ease of implementation, and superior accuracy than others, pedagogical approaches are commonly utilized [17]. RxREN [18], RxNCM [19], BRIANNE [20], and X-TREPAN [21] are some current and successful pedagogical techniques. Among them, the most recent is RxNCM, which extracts rules by reverse engineering a Neural Network by removing inconsequential input neurons and then generates rules using properly classified and misclassified patterns.

The RxNCM [19] algorithm has some inherent drawbacks in its workings. First, the RxNCM algorithm solely employs a sequential feature removal strategy in which a feature that is judged inconsequential is permanently eliminated; this is known as the nesting effect. This indicates that the significance of the feature will not be taken into account in subsequent iterations. As a result, the RxNCM technique does not evaluate the combinations of input neurons rather subsets in order to find the best one, and so its performance suffers. Second, regardless of whether the characteristics are continuous or discrete, the RxNCM computes data ranges for all of them. Because patterns with discrete properties cannot be adequately represented by data ranges, the resulting rules are erroneous in nature. Third, reclassification is used by the RxNCM to update the final ruleset. Though this improves accuracy, the new data range may still contain some overlapping.

Keeping all of these disadvantages in mind, the research objective was to present a new pedagogical rule extraction technique called Comprehensible and Transparent Rule Extraction using Neural Network (CTRENN) to enhance the RxNCM approach. Firstly, the CTRENN enhances the network-pruning phase by pruning the input neurons using backward floating approach. The backward floating approach considers characteristics that have been considered irrelevant for later iterations, eliminating nesting effect. Secondly, CTRENN has distinct approaches for handling discrete and continuous characteristics. For

discrete characteristics, decision trees are used to produce IF-ELSE rules, while for continuous attributes, obligatory data ranges are used to generate rules. Finally, in the rule updating phase, CTRENN employs probabilistic techniques to address the class overlapping dilemma. The probabilistic technique eliminates overlap by changing the higher and lower data ranges based on the class likelihood of each characteristic. This enhances the performance of the extracted final set of rules.

CTRENN's performance is validated using six datasets from the UCI repository [33], and it is revealed that CTRENN provides more accurate and understandable rules than RxREN and RxNCM. The following is how the paper is structured: Sect. 2 discusses some of the significantly associated literatures, Sect. 3 goes over the suggested technique in-depth, Sect. 4 goes over the experimental findings for validating the algorithm, and Sect. 5 concludes the study.

## 2 Literature survey

Neural networks are effective tools for extracting patterns and detecting complex trends in data that may be utilized to generate meaning from difficult or inaccurate data. However, they have the disadvantage of being fundamentally black boxes in nature. There are, however, several techniques for converting neural networks into white boxes by extracting clear rules from them. Several rule-extraction strategies have been developed to uncover the information contained in neural networks. The rule extraction strategies may be classified into three approaches: decompositional, instructional, and eclectic. To take out rules from the network, decompositional algorithms evaluate the weights and activation functions of the hidden layer neurons.

Towell et al. [22] introduced the SUBSET method, which evaluates the incoming weights of hidden and output neurons, takes into account all potential subsets of incoming weights, and identifies all rule combinations bigger than a preset threshold. It was constrained by the ever-increasing number of propositional rules. Lu et al. [23] presented the NeuroRule method, a decompositional approach for extracting oblique classification rules from neural networks with one hidden layer. The RG component of NeuroRule creates rules that cover as many examples of a separate class as feasible with the fewest amounts of characteristics. NeuroLinear, a technique for obtaining oblique decision rules from neural networks, was proposed by [24]. It's a rule extraction technique of decompositional approach with similar stages to NeuroRule [23]; the difference is in data pretreatment. It does not discriminate between discrete and continuous data as input. Gupta et al. [25] introduced an extended analytic rule extraction approach from feed-forward neural network that uses the strength of a Neural Network's connection weights to extract rules. This GLARE algorithm employs the typical network structure and training methods in rule extraction, as well as a direct mapping between input and output nodes to improve comprehensibility.

Odajima et al. [26] suggested a variant decompositional approach for discrete attribute datasets called Greedy Rule Generation (GRG). The discrete hidden layer activation values are subjected to this approach. Because this is a greedy technique, the number of rules created is significantly lower than that of NeuroRule. The instructional strategies attempt to map the relationship between input and output neurons as nearly as possible to how the Neural Network perceives the relationship. One such instructional approach is the BRI-ANNE technique introduced by [20], it has a significant benefit over the previously stated algorithms in that it does not require discretization and can function with continuous data.

Craven and Shavlik [21] presented TREPAN, a method for extracting rules from Neural Networks in the form of decision trees. During the Neural Network's learning phase, this method queries the network to find the class patterns. Biswas et al. [19] introduced the RxNCM method, which is an enhancement on RxREN[18] algorithm. To produce the rules, the RxREN employs misclassified patterns after deleting inconsequential neurons. The RxNCM algorithm improves on this by generating rules utilizing both the misclassified and properly classified algorithms.

There are also eclectic techniques that combine elements of a decompositional approach with elements of a pedagogical approach. The FERNN algorithm was proposed by [27]. It is an eclectic rule extraction approach that discovers valuable hidden units by utilizing C4.5 to locate information gain. Final rules are created by recognizing and weighting relevant input connections to relevant hidden units. The Rex-CGA approach was proposed by [28]. This strategy works with several hidden layer layers. To produce rules, the Rex-CGA uses CGA to locate clusters of activation values in the hidden layers.

Iqbal [29] developed Hierarchical and Eclectic Rule Extraction through Tree Induction and Combination (HERETIC) which utilizes DT to produce rules from individual nodes in a network and then combines the rules generated by all nodes to construct final rules. To create rules, Fast Extraction of Rules from Neural Networks(FERNN) [27] detects the significant hidden neurons and significant input-hidden connections of a fully connected trained single hidden layer network. Jivani et al. [17] contrasted decompositional, pedagogical, and eclectic rule extraction methods. Using criteria such as network architecture, efficiency, extracted rules, and accuracy, they demonstrated that the pedagogical method is computationally quicker than both decompositional and eclectic strategies while retaining a relatively high level of accuracy.

### 3 Proposed CTRENN

Figure 1 illustrates the algorithm's framework. The method distinguishes between discrete and continuous properties. It produces rules with discrete and contributing qualities one at a time, merging them if accuracy increases. If only discrete characteristics are provided, the algorithm creates rules using only those attributes, and if only continuous attributes are present, the method generates rules using only those attributes. Notations used in the following part has been described in Table 1.

The CTRENN method is divided into seven primary phases that includes- *Optimal Network Architecture, Network Pruning, Attribute Separation, Data Range Calculation, Rule Construction, Rule Pruning, and Rule Updating*. The ideal network architecture is determined in the first step. During the network pruning step, the trained neural network's unimportant input neurons are deleted. Following trimming, the qualities are classified as discrete or continuous. In the data range calculation step, the input data range of each relevant continuous attribute for classification is calculated. The rule creation step creates classification rules for discrete attributes (if present) and continuous attributes (if present) using the data ranges acquired in the previous phase, and combines both types of rules if both are available. During the rule trimming phase, these rules are trimmed to eliminate inconsequential ones. At last, the rule update process fine-tunes the rules by changing the data ranges. The following sections explain each phase of the proposed CTRENN algorithm:

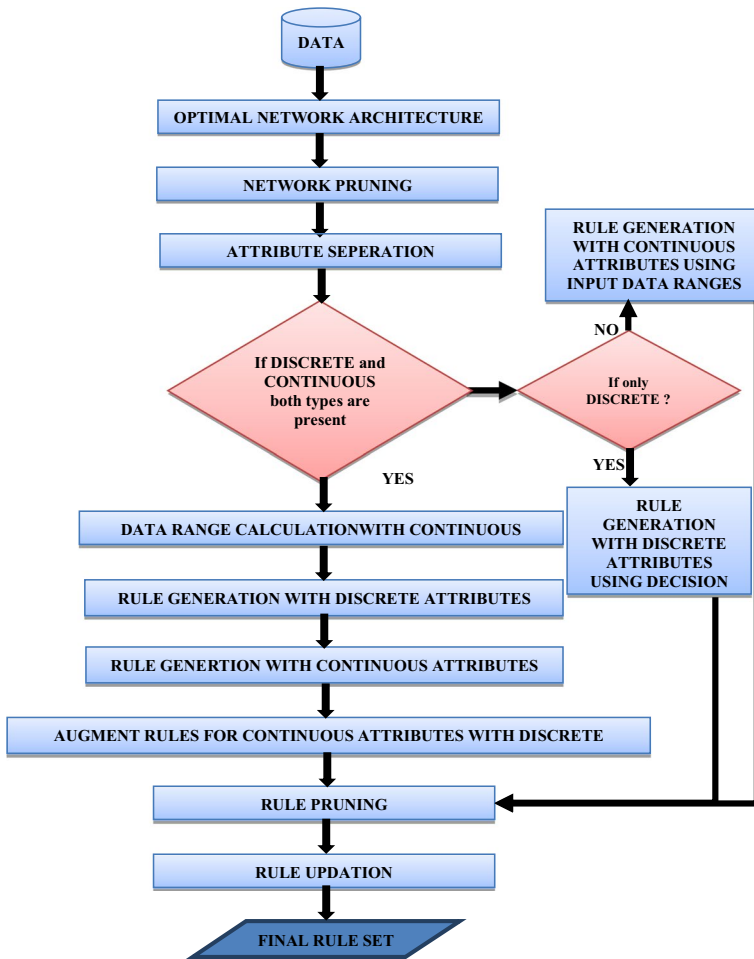


Fig. 1 Flowchart of CTRENN

### 3.1 Optimal network architecture

For rule extraction, a Back Propagation Neural Network (BPNN) with one hidden layer of  $h$  neurons is utilized. The number  $h$  is determined depending on the network’s Mean Square Error(MSE). The network topology is modified from  $l + 1$  to  $2 * l$  hidden neurons, with  $l$  being the number of input neurons, and the structure with the lowest MSE is selected for further processing.

### 3.2 Network pruning

Using the backward floating approach, the CTRENN eliminates irrelevant input neurons from the network. This backward floating approach increases the efficiency of the

**Table 1** Notations used in this article

---

T: Set of classified examples by a Neural Network on a given training set

---

l: Number of input neurons

h: Number of hidden layer neurons

n: Number of output neurons

 $Acc_o$ : Accuracy of a trained network on validation dataset

A: Set of input neurons in network

B: Set of insignificant input neurons

 $Acc_p$ : Accuracy of a pruned network on validation dataset

m: Number of input neurons in the pruned network

 $l_i$ :  $i$  th neuron in the input layer $C_k$ :  $k$  th target class of a dataset $err_i$ : Number of incorrectly classified examples by the trained network without  $l_i$  where  $i \in [1, m]$  $E_i$ : Incorrectly classified examples by the trained network without  $l_i$  $P_i$ : Properly classified examples with only input neuron  $l_i$ UEP <sub>$i$</sub> : Union of  $P_i$  and  $E_i$ ep <sub>$i$</sub> : Total number of examples in UEP <sub>$i$</sub>  for significant input  $l_i$  of pruned networkcep <sub>$ik$</sub> : Number of examples in UEP <sub>$i$</sub>  of class  $C_k$  for significant attribute  $l_i$ DRM <sub>$ik$</sub> : Data Range Matrix $L_{ik}$ : Lower Range of feature  $i$  in DRM <sub>$ik$</sub>  $U_{ik}$ : Upper Range of feature  $i$  in DRM <sub>$ik$</sub>  $min_{ik}$ : Lower Range after rule updation $max_{ik}$ : Upper Range after rule updation $R_k$ : Rule set for class  $k$  $cn_j$ :  $j$  th condition in  $R_k$  where  $j \in [1, m]$ 

D: Set of insignificant rule conditions neurons

 $Acc_r$ : Accuracy of initial rule set  $R_k$  $Acc_j$ : Accuracy of rule set on removal of  $cn_j$ 

final set of rules extracted by making the feature/rule set more trustworthy by taking into account all possible combinations of features/rules. After eliminating the  $i^{th}$  input neurons, CTRENN calculates the number of misclassified cases  $err_i$  for all input neurons  $l_i$ . Here ' $i$ ' ranges in between 1 to number of neurons. A temporary short-term pruned network is produced by removing the input neuron with the lowest  $err_i$  value for the network. This temporary trimmed network's accuracy  $Acc_p$  is determined.  $Acc_p$  is compared to the starting network's current accuracy  $Acc_o$ . If  $Acc_p > Acc_o$ , the program treats this temporarily trimmed network as the pruned network and sets  $Acc_o = Acc_p$ . The pruned input neuron is removed from set **A** of the network and placed in set **B** for later consideration.

CTRENN successively adds the removed input neurons (in set **B**), to the pruned network (set **A**). If the new accuracy ( $Acc_p$ ) is strictly greater ( $Acc_p > Acc_o$ ) than the current accuracy then that input neuron is added back into the network and the accuracy is updated ( $Acc_o = Acc_p$ ). The similar procedure is executed for all the remaining input neurons.

### 3.3 Attribute separation

The significant attributes that remain after the network pruning is separated into two sets, continuous and discrete. Let  $\mathbf{C}$  be the set of continuous attributes and  $\mathbf{D}$  be the set of discrete attributes. The set  $\mathbf{C}$  of continuous attributes are used in the data range computation phase.

### 3.4 Data range calculation

If any continuous attributes remain in the pruned network, this phase is executed. By evaluating the misclassified patterns  $E_i$  in the absence of each  $I_i$  and the properly classified patterns  $P_i$  in the presence of each  $I_i$ , the CTRENN algorithm learns the functioning and relevance of each major continuous input neuron  $I_i$ . CTRENN organizes the examples in set  $UEP_i$  with regard to each target class  $C_k$  and then finds the number of examples  $cep_{ik}$  in each class to obtain the necessary data range of  $I_i$  for each class  $C_k$ . The resulting matrix is known as a data length matrix. The number of examples in the set  $UEP_i$  for the input neuron  $I_i$  is  $ep_i$ . The value of  $k$  is in the range  $[1, n]$ .

All of the characteristics may well not be required for categorizing patterns in all of the  $n$  classes, i.e., for classifying patterns in all of the  $n$  classes, a certain attribute may not be important. As a result, the algorithm chooses data ranges for those attributes which meet the following criteria(1)

$$cep_{ik} > \alpha * ep_i \text{ Forclass } k, \text{ where } \alpha \in [0.1, 0.5] \tag{1}$$

The algorithm constructs a Data Range Matrix (DRM) by finding the lower range  $L_{ik}$  and upper range  $U_{ik}$  of data for each attribute  $I_i$  in class  $C_k$  if  $cep_{ik} > \alpha * ep_i$ . The data range matrix is defined as  $DRM$  with an order  $m * n$  as shown in Fig. 2. Each element of

	$C_1$	$C_2$	$C_3$	...	$C_n$
$I_1$	$[L_{11}, U_{11}]$	$[L_{12}, U_{12}]$	$[L_{13}, U_{13}]$	...	$[L_{1n}, U_{1n}]$
$I_2$	$[L_{21}, U_{21}]$	$[L_{22}, U_{22}]$	$[L_{23}, U_{23}]$	...	$[L_{2n}, U_{2n}]$
$I_3$	$[L_{31}, U_{31}]$	$[L_{32}, U_{32}]$	$[L_{33}, U_{33}]$	...	$[L_{3n}, U_{3n}]$
...	...	...	...	...	...
$I_m$	$[L_{m1}, U_{m1}]$	$[L_{m2}, U_{m2}]$	$[L_{m3}, U_{m3}]$	...	$[L_{mn}, U_{mn}]$

Fig. 2 Data Range Matrix

**DRM** is represented by lower and upper data ranges of the corresponding attributes and classes i.e.,  $L_{ik}$  and  $U_{ik}$ .  $DRM_{ik}$  is calculated using Eq. (2).

$$\left. \begin{array}{l} \text{If } cep_{ik} > \alpha * ep_i, DRM_{ik} = [L_{ik}, U_{ik}] \\ \text{Else, } DRM_{ik} = 0 \end{array} \right\} \quad (2)$$

### 3.5 Rule construction

The rule construction phase is composed of 2 parts if both discrete and continuous types of attributes are there in the network. The first part deals with discrete attributes and the second part deals with continuous attributes. In the first part, rules are constructed with the discrete attributes using C4.5 decision tree (DT). Next, for each rule in the rule-set, the rules with the continuous attribute ( $R_k$ ) are included and the accuracy is checked. If the accuracy increases then the rule with continuous attributes is kept in the final set. The working algorithm for rule construction phase if the pruned network contains both discrete and continuous attributes is given in Table 2.

If only discrete attribute is present, the algorithm only generates rules with discrete attributes using DT and goes rule pruning phase. Similarly, if only continuous attributes remains, the algorithm only generates rules with continuous attributes and goes to rule pruning phase.

### 3.6 Rule pruning

Redundant conditions from the rule-set are removed in this step. A condition  $cn_j$  from initial rule  $R_k$  is removed first and then the CTRENN algorithm calculates the current accuracy  $Acc_j$ . If  $Acc_j > Acc_r$  the removed condition is added to set **D**, considering  $Acc_r$  as the initial accuracy. Next removed conditions in set **D** are sequentially added back one by one. If the

**Table 2** Rule construction algorithm for discrete and continuous attributes

Step 1	Using only the discrete attributes $D$ create a DT using C4.5
Step 2	Generate a set of classification rules $R$ using the DT Let $Acc_R$ be accuracy for rule set $R$
Step 3	For each rule $R_k$ (continuous set) in $R$ do steps 4 to 5 Let $C_k$ be the class $R_k$ predicts Go to Step 4
Step 4	For $i = 1$ to $m$ Step 4.1. If $cep_{ik} > \alpha * ep_i$ then $cn_j = (data(l_i) \geq L_{ik} \wedge data(l_i) \leq U_{ik})$ Step 4.1.1. If $j == 1$ , then $cn = cn_j$ Else, $cn = cn \wedge cn_j$ Step 4.1.2. $j = j + 1$ Write the continuous attribute rule in if-then rule format and add it to $R_k$ Let new accuracy be $Acc_{NewR}$
Step 5	If $(Acc_{NewR} > Acc_R)$ Keep continuous attribute rule in $R_k$ Else Remove continuous attribute rule from $R_k$



**Table 3** Description of the datasets

Dataset	Shape of Dataset	# of Patterns	# of Attributes	# of Classes
Bank Marketing	4119 × 16	4119	16	2
Census	48,842 × 14	31,655	14	2
Heart Disease	303 × 13	270	13	2
German Credit	1000 × 20	345	20	2
Breast Cancer Wisconsin(Original)	699 × 9	683	9	2
Australian Credit	690 × 14	690	14	2

# denotes Numbers

**Table 4** Optimal Network Architecture

Dataset	Optimal Architecture
Bank Marketing	16–23-1
Census	14–19-1
Heart Disease	13–21-1
German Credit	20–33-1
Breast Cancer	9–13-1
Australian Credit	14–20-1

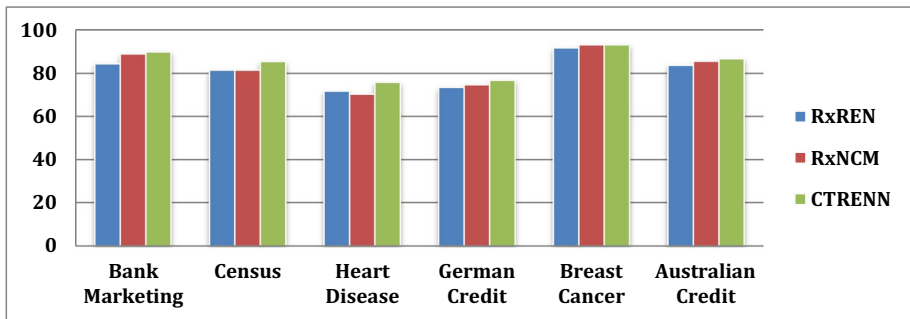
**accuracy**<sub>new</sub> is strictly greater than the **accuracy**<sub>current</sub> then that condition is added back into the network permanently and the accuracy is updated.

### 3.7 Rule update

There may be overlap between various classes in the data range generated for the attribute. CTR-ENN's rule updating phase enhances accuracy by employing a probabilistic strategy to shift the upper and lower data range. For each and every rule, a condition  $-cn_j$  represents an attribute for which there is one lower limit value denotes by (L) and one upper limit value denotes by (U). The overlap happens if the data range of one category intersects with the data range of another. In the case of discrete attributes, the CTRENN considers each value and in the case of continuous attributes a defined range of values. CTRENN determines the probability ( $P_k$ , where  $k \in [1, n]$ ) for each attribute value belonging to more than one class if the data spans for both classes. If  $P_k > 1/n$ , where  $n =$  number of classes, then it is allocated to the data range of that attribute for class  $k$ . Let the new minimum and maximum values of the attribute  $I_i$  for class  $C_k$  be  $min_{ik}$  and  $max_{ik}$ . New  $DRM_{ik} = [min_{ik}, max_{ik}]$ . The new rule set's classification accuracy is  $R_{newacc}$ . Before updating rule-set R, the algorithm alters the  $cn_j$  if  $R_{newacc} \geq R_{acc}$ , where  $R_{acc}$  is the classification accuracy. For each attribute, the rule change is repeated.

**Table 5** Comparison of accuracy in percentage for 10-fold CV

Dataset	RxREN	RxNCM	CTRENN
Bank Marketing	84.27	88.88	<b>89.84</b>
Census	81.48	81.48	<b>85.33</b>
Heart Disease	71.78	70.37	<b>75.87</b>
German Credit	73.50	74.67	<b>76.76</b>
Breast Cancer	91.73	93.16	<b>93.16</b>
Australian Credit	83.64	85.50	<b>86.67</b>

**Fig. 3** Graphical Comparison of accuracy

## 4 Experimental Results

Six authentic benchmark datasets from the UCI repository [33] were utilized to evaluate the proposed CTRENN method. Among the datasets, Breast Cancer was an example of continuous dataset, while the others were of mixed mode. Table 3 has a comprehensive overview of the six datasets.

Mean Square Error (MSE) of the network were utilized to find the optimal architecture with hidden nodes ranging from  $(h = 1 + 1)$  to  $(h = 2 * 1)$ , where  $h$  = number of hidden layer neurons and  $1$  = number of input neurons plus one bias. Table 4 shows the optimal architectures for all datasets.

The following criteria's were utilized to assess the performance of the model: tenfold Cross Validation (CV) accuracy, the average number of extracted rules (G.C.), the average number of antecedents (L.C.), the Recall, the Precision, the F-Measure and the FP-Rate. The tenfold CV method is widely regarded to minimize biases associated in validation. Two new pedagogical rule extraction models are used here to show the comparative performance of the CTRENN. Table 5 demonstrates the comprehensive comparison of CTRENN with RxREN and RxNCM using tenfold CV accuracy for each of the datasets. Figure 3 shows the graphical comparison of accuracy between the algorithms. The results show that CTRENN performs better than RxREN and RxNCM. Formulations have been stated in the following section.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad \text{FP - Rate} = \frac{FP}{TN+FP}$$

$$\mathbf{F - measure} = \frac{2 * (\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

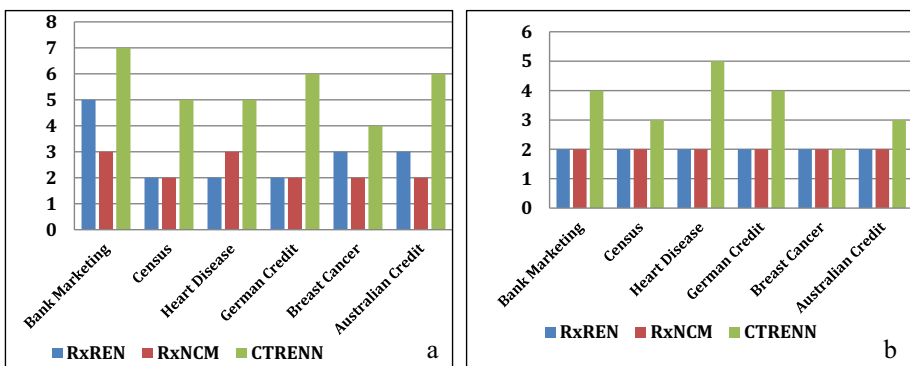
The number of extracted rules gives a significance of the Global Comprehensibility (G.C.) of a rule set. This measure is useful in understanding the depth of the knowledge in the hidden layers of the neural networks. The number of antecedents gives a significance of the Local Comprehensibility (L.C.) of a rule set. This gives a measure of the significance of the attributes present in a rule. Table 6 shows the comparison between CTRENN, RxNCM and RxREN, with respect to L.C. and Table 7 shows comparison in terms of G.C. Comprehensibility is enhanced when the number of attributes is minimal. Figure 4a and 4b

**Table 6** Comparison of Local Comprehensibility (L.C.) for tenfold CV

Dataset	RxREN	RxNCM	CTRENN
Bank Marketing	5	3	7
Census	2	2	5
Heart Disease	2	3	5
German Credit	2	2	6
Breast Cancer	3	2	4
Australian Credit	3	2	6

**Table 7** Comparison of Global Comprehensibility (G.C.) for tenfold CV

Dataset	RxREN	RxNCM	CTRENN
Bank Marketing	2	2	4
Census	2	2	3
Heart Disease	2	2	5
German Credit	2	2	4
Breast Cancer	2	2	2
Australian Credit	2	2	3



**Fig. 4** a Graphical Comparison of L.C. b Graphical Comparison of G.C

**Table 8** Comparison of Recall for tenfold CV

Dataset	RxREN	RxNCM	CTRENN
Bank Marketing	0.7953	0.8325	<b>0.8634</b>
Census	0.8133	0.8133	<b>0.8412</b>
Heart Disease	0.6567	0.6363	<b>0.6966</b>
German Credit	0.7025	0.7306	<b>0.7433</b>
Breast Cancer	0.9067	0.9183	<b>0.9183</b>
Australian Credit	0.8333	0.8452	<b>0.8533</b>

**Table 9** Comparison of Precision for tenfold CV

Dataset	RxREN	RxNCM	CTRENN
Bank Marketing	0.7704	0.8167	<b>0.8435</b>
Census	0.8041	0.8041	<b>0.8255</b>
Heart Disease	0.6539	0.6363	<b>0.7</b>
German Credit	0.7328	0.7415	<b>0.7481</b>
Breast Cancer	0.9188	0.9365	<b>0.9565</b>
Australian Credit	0.8132	0.8398	<b>0.8511</b>

**Table 10** Comparison of F-Measure for tenfold CV

Dataset	RxREN	RxNCM	CTRENN
Bank Marketing	0.7826	0.8245	<b>0.8533</b>
Census	0.8086	0.8086	<b>0.8332</b>
Heart Disease	0.6552	0.6363	<b>0.6982</b>
German Credit	0.7173	0.7360	<b>0.7456</b>
Breast Cancer	0.9127	0.9273	<b>0.9371</b>
Australian Credit	0.8231	0.8424	<b>0.8521</b>

**Table 11** Comparison of FP-Rate for tenfold CV

Dataset	RxREN	RxNCM	CTRENN
Bank Marketing	0.2366	0.1437	<b>0.1325</b>
Census	0.1717	0.1717	<b>0.0255</b>
Heart Disease	0.2347	0.25	<b>0.1747</b>
German Credit	0.5	0.4251	<b>0.3714</b>
Breast Cancer	0.0254	0.0267	0.0267
Australian Credit	0.5666	0.25	<b>0.2067</b>

shows the graphical comparison of L.C. and G.C respectively among the algorithms. The results show that for all datasets CTRENN has both L.C. and G.C. higher than the other algorithms. Perhaps, the reason for this is that CTRENN uses separate rule generation procedures for discrete and continuous attributes/features. The higher number of rules results in a higher predictive accuracy but also results in lower comprehensibility.

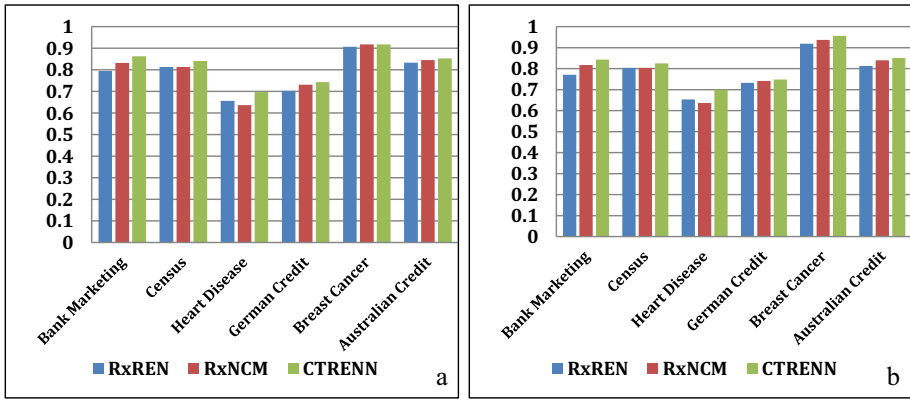


Fig. 5 a Graphical Comparison of Recall. b Graphical Comparison of Precision

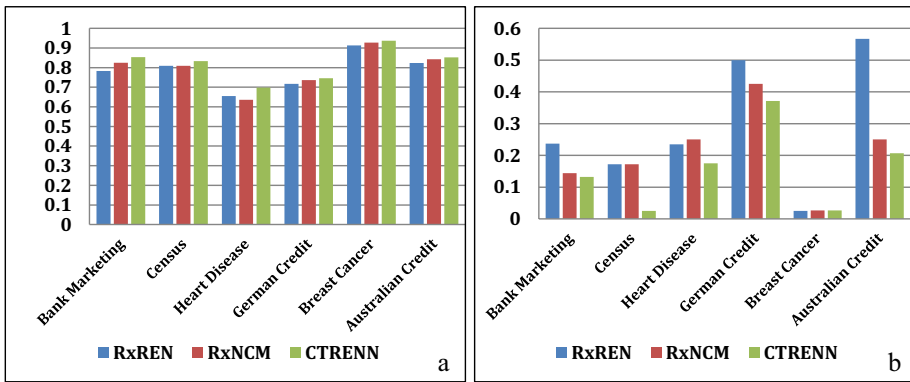


Fig. 6 a Graphical Comparison of F-Measure Fig. 6b. Graphical Comparison of FP-Rate

Table 8, 9, 10, and 11 shows the comparison between CTRENN, RxNCM and RxREN in terms of Recall, Precision, F-Measure and FP-Rate. Fig. 5a and b shows the graphical comparison of Recall and Precision among the algorithms respectively. Fig. 6a and b shows the graphical comparison of F-Measure and FP-Rate among the algorithms respectively.

The experimental results obtained above show that CTRENN is able to obtain better predictive accuracy than the other pedagogical algorithms – RxREN and RxNCM. It can be noted that RxREN and RxNCM do not distinguish between discrete and continuous attributes/features during rule generation. This leads to them always to have exactly 2 rules per rule set. In case of the CTRENN discrete and continuous attributes/features are separated for rule generation. This makes the classification rules produce higher predictive accuracy but suffers from having more rules per rule set. This leads to CTRENN to have worse comprehensibility than that of RxREN and RxNCM. Here, the compromise is between accuracy and comprehensibility.

All the results presented above show that the rules generated by the proposed algorithm are accurate and justifiable. And also, the rules generated are transparent enough to distinguish between different types of attributes making a decision.

## 5 Conclusion

The rule extraction algorithm CTRENN transforms a neural network from a black box to a white box structure by extracting the network's accumulated information as human-understandable rules. As the CTRENN algorithm follows the pedagogical approach to rule extraction, CTRENN extracts classification rules by utilizing the connection between the input and output layer neurons. The CTRENN improves upon the RxNCM algorithm by introducing three novelties. The novelties of the CTRENN algorithm lie in the network-pruning and rule-pruning phase, attribute separation into discrete and continuous sets, and the rule updating phase. To enhance the performance, the CTRENN algorithm utilizes the backward floating search strategy to prune the network and the rules. Moreover, Nesting effects which occur in the sequential feature selection approach are avoided using the backward floating technique. CTRENN uses separate methods for dealing with discrete and continuous attributes. For discrete attributes, IF-ELSE rules are generated using decision trees and for continuous attributes, the rules are generated using the corresponding attributes data range. To avoid the inherent overlap of classes, the CTRENN adopts a probabilistic method in rule updating phase. CTRENN eliminates overlap by changing upper and lower data-ranges based on the class likelihood of each attribute value.

Six real datasets from the UCI repository are utilized to validate the algorithm's performance. The results suggest that the proposed method is successful in terms of the accuracy and other performance metrics. When compared to the RxREN and RxNCM algorithms, the suggested CTRENN method generates more accurate but less comprehensible rules. Hence, the trade-off is between accuracy and comprehensibility. The presented rule extraction approach shows to be an effective tool for comprehending the decisions produced by a neural network in a human-understandable format. The algorithm has a wide range of applications, including medical diagnostics, financial issues, and more. To increase the comprehensibility of the ruleset, additional changes to the algorithm can be used to reduce the amount of produced rules. Data Range Calculation, Rule Construction, Rule Pruning, and Rule Updating process can be improvised to get better transparency with higher accuracy and comprehensibility. In this respect, further studies are required to make the model better in terms of comprehensibility with higher accuracy.

**Acknowledgements** This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

**Data availability** The authors declare that the data supporting the findings of this study are available publicly, UCI repository [<http://archive.ics.uci.edu/ml>].

## Declarations

**Conflict of interest** The authors declare no competing interests.

## References

1. Han J, Pei J, Kamber M (2011) Data mining: concepts and techniques. Elsevier. <https://doi.org/10.1016/C2009-0-61819-5>
2. Midha N, Singh V (2015) A Survey on Classification Techniques in Data Mining. *Int J of Comp Sci Management Stud* 16(1):9–12


3. Mann AK, Kaur N (2013) Survey paper on clustering techniques. *Int J Sci , Eng Technol Res* 2(4):803–806
4. Shridhar M, Parmar M (2017) Survey on association rule mining and its approaches. *Int J Comp Sci Eng (IJCSE)* 5(3):129–135
5. Sharma AK, Sahni S (2011) A comparative study of classification algorithms for spam email data analysis. *Int J Comp Sci Eng* 3(5):1890–1895
6. Kaviani P, Dhotre S (2017) Short survey on naive bayes algorithm. *Int J of Adv Eng Res Develop* 4(11):607–611
7. Cohen S, Rokach L, Maimon O (2007) Decision-tree instance-space decomposition with grouped gain-ratio. *Inf Sci* 177(17):3592–3612. <https://doi.org/10.1016/j.ins.2007.01.016>
8. Mashayekhi M, Gras R (2015) Rule extraction from random forest: the RF+HC methods. In: Barbosa D, Milius E (eds) *Advances in artificial intelligence. Canadian AI 2015. Lecture notes in computer science*, vol 9091. Springer, Cham. [https://doi.org/10.1007/978-3-319-18356-5\\_20](https://doi.org/10.1007/978-3-319-18356-5_20)
9. Kaikhah K, Doddameti S (2006) Discovering trends in large datasets using neural networks. *Appl Intell* 24(1):51–60. <https://doi.org/10.1007/s10489-006-6929-9>
10. Caruana R, Niculescu-Mizil A (2006) An empirical comparison of supervised learning algorithms. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp 161–168. <https://doi.org/10.1145/1143844.1143865>
11. Dam HH, Abbass HA, Lokan C, Yao X (2007) Neural-based learning classifier systems. *IEEE Trans Knowl Data Eng* 20(1):26–39. <https://doi.org/10.1109/TKDE.2007.190671>
12. Mantas CJ, Puche JM, Mantas JM (2006) Extraction of similarity based fuzzy rules from artificial neural networks. *Int J Approximate Reasoning* 43(2):202–221. <https://doi.org/10.1016/j.ijar.2006.04.003>
13. Andrews R (1995) Inserting and extracting knowledge from constrained error back-propagation networks. In: *Proceedings of the 6th Australian Conference on Neural Networks*. NSW
14. Craven MW, Shavlik JW (2014) Understanding neural networks via rule extraction and pruning. In: *Proceedings of the 1993 Connectionist Models Summer School*. Psychology Press, pp 184–191
15. Botari T, Izbicki R, de Carvalho ACP (2020) Local interpretation methods to machine learning using the domain of the feature space. In: Cellier P, Driessens K (eds) *Machine learning and knowledge discovery in databases. ECML PKDD 2019. Communications in computer and information science*, vol 1167. Springer, Cham. [https://doi.org/10.1007/978-3-030-43823-4\\_21](https://doi.org/10.1007/978-3-030-43823-4_21)
16. Bologna G, Hayashi Y (2018) A comparison study on rule extraction from neural network ensembles, boosted shallow trees, and SVMs. *Appl Comput Intell Soft Comput* 2018:1–20. <https://doi.org/10.1155/2018/4084850>
17. Jivani K, Ambasana J, Kanani S (2014) A survey on rule extraction approaches based techniques for data classification using neural network. *Int J Futuristic Trends Eng Technol* 1(1):4–7
18. Augusta MG, Kathirvalavakumar T (2012) Reverse engineering the neural networks for rule extraction in classification problems. *Neural Process Lett* 35(2):131–150. <https://doi.org/10.1007/s11063-011-9207-8>
19. Biswas SK, Chakraborty M, Purkayastha B, Roy P, Thounaojam DM (2017) Rule extraction from training data using neural network. *Int J Artif Intell Tools* 26(03):1750006. <https://doi.org/10.1142/S0218213017500063>
20. Sestito S (1992) Automated knowledge acquisition of rules with continuously valued attributes. In: *Proceedings of the 12th International Conference on Expert Systems and their Applications*
21. Craven M, Shavlik J (1995) Extracting tree-structured representations of trained networks. *Adv Neural Inf Process Syst* 8
22. Towell GG, Shavlik JW (1993) Extracting refined rules from knowledge-based neural networks. *Mach Learn* 13(1):71–101. <https://doi.org/10.1007/BF00993103>
23. Lu H, Setiono R, Liu H (2017) NeuroRule: a connectionist approach to data mining. arXiv preprint arXiv:1701.01358. <https://doi.org/10.48550/arXiv.1701.01358>
24. Setiono R, Liu H (1997) NeuroLinear: From neural networks to oblique decision rules. *Neurocomputing* 17(1):1–24. [https://doi.org/10.1016/S0925-2312\(97\)00038-6](https://doi.org/10.1016/S0925-2312(97)00038-6)
25. Gupta A, Park S, Lam SM (1999) Generalized analytic rule extraction for feedforward neural networks. *IEEE Trans Knowl Data Eng* 11(6):985–991. <https://doi.org/10.1109/69.824621>
26. Odajima K, Hayashi Y, Tianxia G, Setiono R (2008) Greedy rule generation from discrete data and its use in neural network rule extraction. *Neural Netw* 21(7):1020–1028. <https://doi.org/10.1016/j.neunet.2008.01.003>
27. Setiono R, Leow WK (2000) FERNN: An algorithm for fast extraction of rules from neural networks. *Appl Intell* 12(1–2):15–25. <https://doi.org/10.1023/A:1008307919726>

28. Hruschka ER, Ebecken NF (2006) Extracting rules from multilayer perceptrons in classification problems: A clustering-based approach. *Neurocomputing* 70(1–3):384–397. <https://doi.org/10.1016/j.neucom.2005.12.127>
29. Al Iqbal MR (2012) Eclectic rule extraction from neural networks using aggregated decision trees. In: 2012 7th International Conference on Electrical and Computer Engineering. IEEE, pp 129–132. <https://doi.org/10.1109/ICECE.2012.6471502>
30. Bhattacharya A, Parui, SK, Biswas SK, Mandal A (2023) An empirical study on credit risk assessment using ensemble classifiers. In: Chakraborty B, Biswas A, Chakrabarti A (eds) *Advances in data science and computing technologies. ADSC 2022. Lecture notes in electrical engineering*, vol. 1056. Springer, Singapore. [https://doi.org/10.1007/978-981-99-3656-4\\_16](https://doi.org/10.1007/978-981-99-3656-4_16)
31. Bhattacharya A, Biswas SK, Mandal A (2023) Credit risk evaluation: a comprehensive study. *Multimed Tools Appl* 82:18217–18267. <https://doi.org/10.1007/s11042-022-13952-3>
32. Payrovnaziri SN, Chen Z, Rengifo-Moreno P, Miller T, Bian J, Chen JH, ... He Z (2020) Explainable artificial intelligence models using real-world electronic health record data: a systematic scoping review. *J Am Med Inform Assoc* 27(7):1173–1185. <https://doi.org/10.1093/jamia/ocaa053>
33. Dua D, Graff C (2019) UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. University of California, School of Information and Computer Science, Irvine

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

## Authors and Affiliations

Saroj Kr. Biswas<sup>1</sup> · Arijit Bhattacharya<sup>2</sup>  · Abhinaba Duttachoudhury<sup>1</sup> · Manomita Chakraborty<sup>3</sup> · Akhil Kumar Das<sup>2</sup>

✉ Arijit Bhattacharya  
barijit@hotmail.com

Saroj Kr. Biswas  
bissarojkum@yahoo.com

Abhinaba Duttachoudhury  
ad.chaudhuri1995@gmail.com

Manomita Chakraborty  
mou.look@gmail.com

Akhil Kumar Das  
dasakhi@gmail.com

<sup>1</sup> NIT, Silchar, Assam 788 010, India

<sup>2</sup> Gour Mahavidyalaya, Malda, West Bengal 732142, India

<sup>3</sup> School of Computer Science and Engineering, VIT-AP University, Amaravathi, India