ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230274

Neural Network-Based Rule Models with Truth Tables

Adrien Benamira*, Tristan Guérand*, Thomas Peyrin and Hans Soegeng

Nanyang Technological University, Singapore
*Main contribution

Abstract. Understanding the decision-making process of a machine/deep learning model is crucial, particularly in security-sensitive applications. In this study, we introduce a neural network framework that combines the global and exact interpretability properties of rule-based models with the high performance of deep neural networks.

Our proposed framework, called *Truth Table rules* (TT-rules), is built upon *Truth Table nets* (TTnets), a family of deep neural networks initially developed for formal verification. By extracting the set of necessary and sufficient rules $\mathcal{R}$ from the trained TTnet model (global interpretability), yielding the same output as the TTnet (exact interpretability), TT-rules effectively transforms the neural network into a rule-based model. This rule-based model supports binary classification, multi-label classification, and regression tasks for tabular datasets. Furthermore, our TT-rules framework optimizes the rule set $\mathcal{R}$ into $\mathcal{R}_{opt}$ by reducing the number and size of the rules. To enhance model interpretation, we leverage Reduced Ordered Binary Decision Diagrams (ROBDDs) to visualize these rules effectively.

After outlining the framework, we evaluate the performance of TT-rules on seven tabular datasets from finance, healthcare, and justice domains. We also compare the TT-rules framework to state-of-the-art rule-based methods. Our results demonstrate that TT-rules achieves equal or higher performance compared to other interpretable methods while maintaining a balance between performance and complexity. Notably, TT-rules presents the first accurate rule-based model capable of fitting large tabular datasets, including two real-life DNA datasets with over 20K features. Finally, we extensively investigate a rule-based model derived from TT-rules using the Adult dataset.

1 Introduction

Deep Neural Networks (DNNs) have been widely and successfully employed in various machine learning tasks, but concerns regarding their security and trustworthiness persist. One of the primary issues associated with DNNs, as well as ensemble ML models in general, is their lack of explainability and the challenge of incorporating human knowledge into them due to their inherent complexity [37, 38]. Therefore, there is a significant research focus on achieving global and exact interpretability for these systems, especially in safety-critical applications [4, 19].

In contrast, rule-based models [22], including tree-based models [10], are specifically designed to offer global and exact explanations, providing insights into the decision-making process that yields the same output as the model. However, they generally exhibit lower performance compared to other models like DNNs or ensemble ML model [27]. Additionally, they encounter scalability issues when dealing with large datasets and lack flexibility in addressing various types of tasks, often limited to binary classification [13].

To the best of our knowledge, there is currently no family of DNNs that possesses both global and exact interpretability akin to rule-based models, while also demonstrating scalability on real-life datasets without the need for an explainer. This limitation is significant since explainer methods often provide only local, inexact, and potentially misleading explanations [37, 38, 40].

Our approach. This paper introduces a novel neural network framework that effectively combines the interpretability of rule-based models with the high performance of DNNs. Our framework, called TT-rules, builds upon the advancements made by Benamira *et al.* [9] and Agarwal *et al.* [3]. The latter proposed a neural network architecture that achieves interpretability by utilizing several DNNs, each processing a single continuous input feature, and a linear layer for merging them. The effectiveness of aggregating local features on image datasets to achieve high accuracy has been demonstrated by Brendel *et al.* [14]. Similarly, Agarwal *et al.* [3] showed that aggregating local features on tabular datasets can also yield high accuracy. Furthermore, Benamira *et al.* [9] introduced a new Convolutional Neural Network (CNN) filter function called the Learning Truth Table (LTT) block. The LTT block has the unique property of its complete distribution being computable in constant and practical time, regardless of the architecture. This allows the transformation of the LTT block from weights into an exact mathematical Boolean formula. Since an LTT block is equivalent to a CNN filter, the entire neural network model, known as Truth Table Net (TTnet), can itself be represented as a Boolean formula.

To summarize, while Agarwal *et al.* [3] focused on continuous inputs, and Benamira *et al.* [9] focused on discrete inputs, our approach leverages the strengths of both works to achieve high accuracy while maintaining global and exact interpretability.

Our contributions. To optimize the rule set $\mathcal{R}$, our TT-rules framework employs two post-training steps. Firstly, we automatically integrate *"Don't Care Terms"* (DCT), utilizing human logic, into the truth tables. This reduces the size of each rule in the set $\mathcal{R}$. Secondly, we introduce and analyze an inter-rule correlation score to decrease the number of rules in $\mathcal{R}$. These optimizations, specific to the TT-rules framework, automatically and efficiently transform the set $\mathcal{R}$ into an optimized set $\mathcal{R}opt$ in constant time. We also quantify the trade-offs among performance, the number of rules, and their sizes. At this stage, we obtain a rule-based model from the trained DNN TTnet, which can be used for prediction by adding up the rules in $\mathcal{R}opt$ according to the binary or floating linear layer. To enhance the interpretability of the model, we convert all rule equations into Reduced Ordered Binary Decision Diagrams.

Please contact the corresponding author for any appendices or supplementary material mentioned in the paper.

Our claims. A) The TT-rules framework demonstrates versatility and effectiveness across various tasks, including binary classification, multi-classification, and regression. A-1) Our experiments encompassed five machine learning datasets: Diabetes [21] in healthcare, Adult [21], HELOC [1], and California Housing [32] in finance, and Compas [7] in the justice domain. The results clearly indicate that the TT-rules framework surpasses most interpretable models in terms of Area Under Curve/Root Mean Square Error (AUC/RMSE), including linear/logistic regression, decision trees, generalized linearized models, and neural additive models. A-2) On two datasets, the TT-rules framework performs comparably to XGBoost and DNN models. A-3) We conducted a comparative analysis of the performance-complexity tradeoff between our proposed TT-rules framework and other state-of-the-art rule-based models, such as generalized linearized models [44], RIPPER [17, 18], decision trees (DT)[33], and ORS[45], specifically focusing on binary classification tasks. Our findings demonstrate that the TT-rules framework outperforms all the aforementioned models, except for the generalized linearized models, in terms of the performance-complexity tradeoff.

B) Scalability is a key strength of our model, enabling it to handle large datasets with tens of thousands of features, such as DNA datasets [41, 34, 31], which consist of over 20K features. Our model not only scales efficiently but also performs feature reduction, compressing the initial 20K features of the first DNA datasets [41, 34] into 1K rules, and reducing the 23K features of the second DNA dataset [31] into 9K rules.

C) A distinctive feature of our framework lies in its inherent global and exact interpretability. C-1) To showcase its effectiveness, we provide a concrete use case with the Adult dataset and thoroughly investigate its interpretability. C-2) We explore the potential for incorporating human knowledge into our framework. C-3) Additionally, we highlight how experts can leverage the rules to detect concept shifts, further emphasizing the interpretability aspect of our framework.

Outline. This paper is structured as follows. Section 2 presents a comprehensive literature review on rule-based models. In Section 3, we establish the notations and fundamental concepts that will be utilized throughout the paper. Section 4 offers a detailed analysis of the TT-rules framework, exploring its intricacies and functionalities. In Section 5, we present the experimental results obtained and compare them with the current state-of-the-art approaches. Additionally, we showcase the scalability of our framework and illustrate its applicability through a compelling case study. The limitations of the proposed approach are discussed in Section 6, followed by the concluding remarks in Section 7.

2 Related work

2.1 Classical rule-based models

Rule-based models are widely used for interpretable classification and regression tasks. This class encompasses various models such as decision trees [10], rule lists [39, 6, 20], linear models, and rule sets [28, 17, 18, 35, 44]. Rule sets, in particular, offer high interpretability due to their straightforward inference process [28]. However, traditional rule sets face limitations when applied to large tabular datasets, binary classification tasks, and capturing complex feature relationships. These limitations result in reduced accuracy and limited practicality in real-world scenarios [45, 43]. To overcome these challenges, we leverage the recent work of Benamira *et al.* [9], who proposed an architecture specifically designed to be encoded into CNF formulas [11]. This approach has demonstrated scalability on large datasets like ImageNet and can be extended to multi-label classification tasks. In this study, our objective is to extend Benamira's approach to handle binary and multi-class classification tasks, as well as regression tasks, across a wide range of tabular datasets ranging from 17 to 20K features.

2.2 DNN-based rule models

There have been limited investigations into the connection between DNNs and rule-based models. Two notable works in this area are DNF-net [2] and RRL [43]. DNF-net focuses on the activation function but lacks available code, while RRL specifically addresses classification tasks. Although RRL achieved high accuracy on the Adult dataset, its interpretability raises concerns due to its complex nature, involving millions of terms, and its training process that is time-consuming [43]. Neural Additive Models (NAMs) [3] represent another type of neural network architecture that combines the flexibility of DNNs with the interpretability of additive models. While NAMs have demonstrated superior performance compared to traditional interpretable models, they do not strictly adhere to the rule-based model paradigm and can pose challenges in interpretation, especially when dealing with a large number of features. In this paper, we conduct a comparative analysis to evaluate the performance and interpretability of our TT-rules framework in comparison to NAMs [3].

3 Background

3.1 Rule-based models

3.1.1 Rules format : DNF and ROBDD

Rule-based models are a popular method for generating decision predicates expressed in DNF. For instance, in the Adult dataset [21], a rule for determining whether an individual would earn more than 50K$/year might look like:

$$((\text{Age} > 34) \wedge \text{Maried}) \vee (\text{Male} \wedge (\text{Capital Loss} < 1\text{k/year}))$$

Although a rule is initially expressed in DNF format, a decision tree format is often preferred. To achieve this, the DNF is transformed into its equivalent Reduced Ordered Binary Decision Diagram (ROBDD) graph: a directed acyclic graph used to represent a Boolean function [29, 5, 15, 8].

3.1.2 Infer a set of rule-based model

In a binary classification problem, we are presented with a set of rules $\mathcal{R}$ and a corresponding set of weights $\mathcal{W}$. These rules and weights can be separated into two distinct sets, namely $\mathcal{R}+$ and $\mathcal{W}+$ for class 1, and $\mathcal{R}-$ and $\mathcal{W}-$ for class 0. Given an input I, we can define the rule-based model as follows:

$$Classifier(I, \mathcal{R}) = \begin{cases} 1 & \text{if } S_+(I) - S_-(I) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Here, $S_+(I)$ and $S_-(I)$ denote the scores for class 1 and class 0, respectively. These scores are calculated using the following equations:

$$\begin{cases} S_+(I) = \sum_{(r_+, w_+) \in (\mathcal{R}+, \mathcal{W}+)} w_+ \times \mathbb{1}_{r+(I) \text{ is True}} \\ S_-(I) = \sum_{(r_-, w_-) \in (\mathcal{R}-, \mathcal{W}-)} w_- \times \mathbb{1}_{r-(I) \text{ is True}} \end{cases}$$

where $\mathbb{1}_{r(I)\ \texttt{is True}}$ represents the binary indicator that is equal to 1 if the input I satisfies the rule r, and 0 otherwise. This rule-based model can be easily extended to multi-class classification and regression tasks.

3.1.3 Comparing rule-based models

When comparing rule-based models, it is common to evaluate their quality based on three main criteria. The first is their performance, which can be measured using metrics such as AUC, accuracy, or RMSE. The second criterion is the number of rules used in the model. Finally, the overall complexity of the model is also taken into account, which is given as the sum of the size of each rule, for all rules in the model [22].

3.2 Truth Table net (TTnet)

The paper [9] proposed a new CNN filter function called Learning Truth Table (LTT) block for which one can compute the complete distribution in practical and constant time, regardless of the network architecture. Then, this LTT block is inserted inside a DNN as CNN filters are integrated into deep convolutional neural networks.

3.2.1 Overall LTT design

An LTT block must meet two essential criteria:

- (A) The LTT block distribution must be entirely computable in practical and constant time, regardless of the complexity of the DNN.
- (B) Once LTT blocks are assembled into a layer and layers into a DNN, the latter DNN should be scalable, especially on large-scale datasets such as ImageNet.

To meet these criteria, Benamira *et al.* [9] proposed the following LTT design rules:

1. Reduce the input size of the CNN filter to $n \leq 9$.
2. Use binary inputs and outputs.
3. Ensure that the LTT block function uses a nonlinear function.

As a result, each filter in our architecture becomes a truth table with a maximum input size of 9 bits.

Notations. We denote the f^{th} 1D-LTT of a layer with input size n, stride s, and no padding as Φ_f. Let the input feature with a single input channel $chn_{input} = 1$ be represented as $(v_0 \dots v_{L-1})$, where L is the length of the input feature. We define $y_{i,f}$ as the output of the function Φ_f at position i:

$$y_{i,f} = \Phi_f(v_{i\times s}, v_{i\times s+1}, \dots, v_{i\times s+(n-1)})$$

Following the aforementioned rules (1) and (2), $y_{i,f}$ and $(v_{i\times s}, v_{i\times s+1}, \dots, v_{i\times s+(n-1)})$ are binary values, and $n \leq 9$. As a result, we can express the 1D-LTT function Φ_f as a truth table by enumerating all 2^n possible input combinations. The truth table can then be converted into an optimal (in terms of literals) DNF formula using the Quine–McCluskey algorithm [12] for interpretation.

Example 1: From LTT weights to truth table and DNF. In this example, we consider a pre-trained 1D-LTT Φ_f with input size $n = 4$, a stride of size 1, and no padding. The architecture of Φ_f is given in Figure 1b composed of two CNN filter layers: the first one has parameters W_1 with (input channel, output channel, kernel size, stride) = $(1, 4, 3, 1)$, while the second W_2 with $(4, 1, 2, 1)$. The inputs and outputs of Φ_f are binary, and we denote the inputs as [x_0, x_1, x_2, x_3]. To compute the complete distribution of Φ_f, we generate all $2^4 = 16$ possible input/output pairs, as shown in Figure 1a, and obtain the truth table in Table 1. This truth table fully characterizes the behavior of Φ_f. We then transform the truth table into a DNF using the Quine-McCluskey algorithm [12]. This algorithm provides an optimal (in terms of literals) DNF formula that represents the truth table. The resulting DNF formula for Φ_f can be used to compute the output of Φ_f for any input. Overall, this example demonstrates the applicability of LTT design rules in the construction of DNNs, as it meets both criteria of LTT blocks being computable in constant time and DNN scalability on large datasets.

3.2.2 Overall TTnet design

We integrated LTT blocks into the neural network, just as CNN filters are integrated into a deep convolutional neural network: each LTT layer is composed of multiple LTT blocks and there are multiple LTT layers in total. Additionally, there is a pre-processing layer and a final layer. These two layers provide flexibility in adapting to different applications: scalability, formal verification, and logic circuit design.

4 Truth Table Rules (TT-rules)

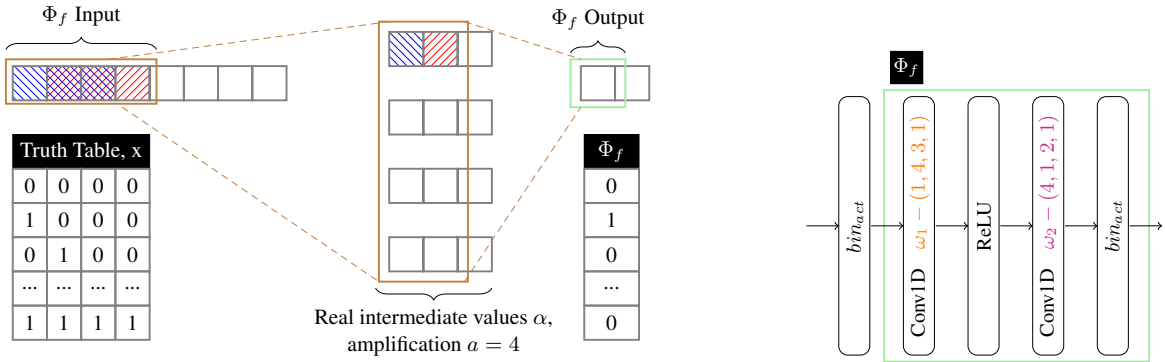
The Truth Table rules framework consists of three essential components. The first step involves extracting the precise set of rules $\mathcal{R}$ once the TTnet has been trained. Next, we optimize $\mathcal{R}$ by reducing the rule's size through *Don't Care Terms* (DCT) injection. At this point, $\mathcal{R}$ is equivalent to the Neural Netwok model: inferring with $\mathcal{R}$ is the same as inferring with the model. Last, we minimize the number of rules using the Truth Table correlation metric. Both techniques serve to enhance the model's complexity while minimizing any potential loss of accuracy.

4.1 From LTT block to set of rules $\mathcal{R}$

General. We now introduce a method to convert Φ_f from the general DNF form into rule set $\mathcal{R}$. In the previous section, we described the general procedure for transforming an LTT block into a DNF logic gate expression. This expression is independent of the spatial position of the feature. This means that we have:

$$\begin{cases} y_{0,f} = \Phi_f(v_0, v_1, \dots, v_{(n-1)})) \\ \dots \\ y_{i,f} = \Phi_f(v_{i\times s}, v_{i\times s+1}, \dots, v_{i\times s+(n-1)})) \\ \dots \\ y_{\lfloor \frac{L-n}{s} \rfloor, f} = \Phi_f(v_{L-n}, v_{L-n+1}, \dots, v_{L-1})) \end{cases}$$

When we apply the LTT DNF expression to a specific spatial position on the input, we convert the DNF into a rule. To convert the general DNF form into a set of rules $\mathcal{R}$, we divide the input into patches and substitute the DNF literals with the corresponding feature names. The number of rules for one filter corresponds to the number of patches: $\lfloor \frac{L-n}{s} \rfloor$. An example of this process is given in Table 1 and one is provided below.



(a) LTT filter Φ_f computation and transformation into a truth table. Φ_f is characterized by weights ω_1 with parameters (input channel, output channel, kernel size, stride) = $(1, 4, 3, 1)$, and $\omega_2 = (4, 1, 2, 1)$. Φ_f input bit-size is 4 (i.e., brown box) since the output feature (i.e., green box) requires 4 input entries (i.e., blue & red hashed boxed).

(b) LTT overview in one dimension. The intermediate values and the weights (ω_1, ω_2) are real and the input/output values are binary. bin_{act}: Heaviside step function defined as $bin_{act}(x) = \frac{1}{2} + \frac{sgn(x)}{2}$ with $x \in \mathbb{R}$.

Figure 1: A Learning Truth Table (LTT) filter example in one dimension.

Table 1: Truth Table of the LTT block Φ_f characterized by the weights W_1 and W_2 with $L = 5$ and binary input feature names [Is the Sex Male? (Male), Did the person go to University? (Go Uni.), Is the person married? (Married), Is the person born in the US? (Born US), Is the person born in the UK? (Born UK)].

x_0	x_1	x_2	x_3	Φ_f
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$$W_1 = \begin{pmatrix} 10 & -1 & 3 \\ 6 & -5 & 4 \\ 4 & 4 & -3 \\ 4 & 4 & 3 \end{pmatrix}$$
$$W_2 = \begin{pmatrix} -5 & 0 & 9 & -5 \\ -5 & 4 & 0 & 0 \end{pmatrix}$$

Φ_f expression in DNF
$x_3 \wedge \overline{x_0} \wedge \overline{x_1} \wedge \overline{x_2}$
Input space
[Male, Go Uni., Married, Born US, Born UK]
$\text{Rule}_0^{\text{DNF}}$
Born US $\wedge$ $\overline{\text{Male}}$ $\wedge$ $\overline{\text{Go Uni.}}$ $\wedge$ $\overline{\text{Married}}$
$\text{Rule}_1^{\text{DNF}}$
Born UK $\wedge$ $\overline{\text{Go Uni.}}$ $\wedge$ $\overline{\text{Married}}$ $\wedge$ $\overline{\text{Born US}}$
$\text{Rule}_{1,reduced}^{\text{DNF}}$
Born UK $\wedge$ $\overline{\text{Go Uni.}}$ $\wedge$ $\overline{\text{Married}}$

Example 2: conversion of DNF expressions to rules. We established the Φ_f expression in DNF form as $x_3 \wedge \overline{x_0} \wedge \overline{x_1} \wedge \overline{x_2}$. To obtain the rules, we need to consider the padding and the stride of the LTT block. Consider the following 5-feature binary input ($L = 5$): [Male, Go Uni., Married, Born in US, Born in UK]. In our case, with a stride at 1 and no padding, we get 2 patches: [Male, Go Uni., Married, Born US] and [Go Uni., Married, Born US, Born UK]. After the substitution of the literal by the corresponding feature, we get 2 rules $\mathcal{R} = \{\text{Rule}_0^{\text{DNF}}, \text{Rule}_1^{\text{DNF}}\}$:

$$\begin{cases} \text{Rule}_{0,f}^{\text{DNF}} = \text{Born US} \wedge \overline{\text{Male}} \wedge \overline{\text{Go Uni.}} \wedge \overline{\text{Married}} \\ \text{Rule}_{1,f}^{\text{DNF}} = \text{Born UK} \wedge \overline{\text{Go Uni.}} \wedge \overline{\text{Married}} \wedge \overline{\text{Born US}} \end{cases}$$

and therefore, the output of the LTT block Φ_f becomes:

$$\begin{cases} y_{0,f} = \text{Rule}_{0,f}^{\text{DNF}}(v_0, v_1, v_2, v_3) \\ y_{1,f} = \text{Rule}_{1,f}^{\text{DNF}}(v_1, v_2, v_3, v_4) \end{cases}$$

We underline the logic redundancy in $\text{Rule}_1^{\text{DNF}}$: if someone is born in the UK, he/she is necessarily not born in the US. We solve this issue by injecting *Don't Care Terms* (DCT) into the truth table as we will see in the next section.

4.2 *Automatic post-training optimizations: from $\mathcal{R}$ to $\mathcal{R}_{opt}$*

In this subsection, we present automatic post-training optimizations that are unique to our model and require the complete computation of the LTT truth table.

4.2.1 *Reducing the rule's size with Don't Care Terms (DCT) injection*

We propose a method for reducing the size of rules by injecting *Don't Care Terms* (DCT) into the truth table. These terms represent situations where the LTT block output can be either 0 or 1 for a specific input, without affecting the overall performance of the DNN. We use the Quine-McCluskey algorithm to assign the optimal value to the DCT and reduce the DNF equations. These DCT can be incorporated into the model either with background knowledge or automatically with the one hot encodings and the Dual Step Function described in the TTnet paper [9].

To illustrate this method, we use Example 2 where we apply human common sense and reasoning to inject DCT into the truth table. For instance, since no one can be born in both the UK and the US at the same time, the literals x_2 and x_3 must not be 1 at the same time for the second rule. By injecting DCT into the truth table as $[0, 0, 1, DCT, 0, 0, 0, DCT, 0, 0, 0, DCT, 0, 0, 0, DCT]$, we obtain the new reduced rule: $\text{Rule}_{1,reduced}^{\text{DNF}} = \text{Born UK} \wedge \overline{\text{Go Uni.}} \wedge \overline{\text{Married}}$. This method significantly decreases the size of the rules while maintaining the same accuracy, as demonstrated in Table 4 in Section 5.

4.2.2 *Reducing the number of rules with Truth Table Correlation metric*

To reduce the number of rules obtained with the TT-rules framework, we introduce a new metric called Truth Table Correlation (TTC). This metric addresses the issue of rule redundancy by measuring

Table 2: Comparison machine learning dataset of our method to Linear/Logistic Regression) [33], Decision Trees (DT) [33], GL [44], NAM [3], XGBoost [16] and DNNs. Results are obtained with a large TT-rules model, without optiizations. Means and standard deviations are reported from 5-fold cross validation.

	Regression (RMSE)	**Binary classification** (AUC)			**Multi-classification** (Accuracy)
continous/binary #	California Housing 8/144 features	Compas 9/17 features	Adult 14/100 features	HELOC 24/330 features	Diabetes 43/296 features
Linear/ log	0.728 ± 0.015	0.721 ± 0.010	0.883 ± 0.002	0.798 ± 0.013	0.581 ± 0.002
DT	0.514 ± 0.017	0.731 ± 0.020	0.872 ± 0.002	0.771 ± 0.012	0.572 ± 0.002
GL	0.425 ± 0.015	0.735 ± 0.013	0.904 ± 0.001	0.803 ± 0.001	NA
NAM	0.562 ± 0.007	0.739 ± 0.010	-	-	-
TT-rules (Ours)	0.394 ± 0.017	0.742 ± 0.007	0.906 ± 0.005	0.800 ± 0.001	0.584 ± 0.003
XGBoost	0.532 ± 0.014	0.736 ± 0.001	0.913 ± 0.002	0.802 ± 0.001	0.591 ± 0.001
DNNs	0.492 ± 0.009	0.732 ± 0.004	0.902 ± 0.002	0.800 ± 0.010	0.603 ± 0.004

the correlation between two different LTT blocks, which may learn similar rules since they are completely decoupled from each other. The idea is to identify and remove redundant rules and keep only the most relevant ones.

The TTC metric is defined as follows:

$$\text{TTC}(y_1, y_2) = \begin{cases} \frac{HW(y_1,\overline{y_2})}{|y_1|} - 1 & \text{if } abs(\frac{HW(y_1,\overline{y_2})}{|y_1|} - 1) > \frac{HW(y_1,\overline{y_2})}{|y_1|} \\ \frac{HW(y_1,y_2)}{|y_1|} & \text{otherwise.} \end{cases}$$

Here, y_1 and y_2 are the outputs of the LTT blocks, $\overline{y_2}$ is the negation of y_2, $|y_1|$ represents the number of elements in y_1, and HW is the Hamming distance function. The Hamming distance between two equal-length strings of symbols is the number of positions at which the corresponding symbols are not equal. The TTC metric varies from -1 to 1. When $TTC = -1$, the LTT blocks are exactly opposite, while they are the same if $TTC = 1$. We systematically filter redundant rules with a threshold correlation of ± 0.9. If the correlation is positive, we delete one of the two filters and give the same value to the second filter. If the correlation is negative, we delete one of the two filters and give the opposite value to the second filter. By using this metric, we can reduce the number of rules and optimize the complexity of the model while minimizing accuracy degradation.

4.3 Overall TT-rules architecture

4.3.1 Pre-processing and final layer

To maintain interpretability, we apply batch normalization and a step function layer consisting of a single linear layer. The batch normalization allows to learn the thresholds for the continuous features (such as the condition YoE > 11 in Fig. 2). We propose two types of training for the final linear layer. The first uses a final sparse binary layer, which forces all weights to be binary and sparse according to a BinMask as in [26]. In order to train without much loss in performance when using the Heaviside step function, Benamira *et al.* [9] adopted the Straight-Through Estimator (STE) proposed by [25]. The second is designed for scalability and employs floating-point weights, which allows to extend the model to regression tasks. To reduce overfitting, a dropout function is applied in the second case.

4.3.2 Estimating complexity before training

In our TT-rules framework, the user is unable to train a final rule-based model with a fixed and pre-selected complexity. However, the complexity can be estimated. The number of rules is determined by multiplying the number of filters F by the number of patches $\lfloor \frac{L-n}{s} \rfloor$. The complexity of each rule is based on the size of the function n, and on average, we can expect $n2^{n-1}$ Boolean gates per rule, before DCT injection. Therefore, the overall complexity is given by $n \times 2^{n-1} \times \lfloor \frac{L-n}{s} \rfloor \times F$.

4.3.3 Training and extraction time

Training. Compared to other rule-based models, our architecture scales well in terms of training time. The machine learning tabular dataset can be trained in 1-5 minutes for 5-fold cross-validation. For large DNA tabular datasets, our model can be trained in 45 minutes for 5-fold cross-validation, which is not possible with other rule-based models such as GL and RIPPER.

Extraction time for $\mathcal{R}_{opt}$. Our model is capable of extracting optimized rules at a fast pace. Each truth table can be computed in 2^n operations, where $n \leq 9$ is in terms of complexity. In terms of time, our model takes 7 to 17 seconds for Adult [21], 7 to 22 seconds for Compas [7], and 20 to 70 seconds for Diabetes [21].

5 Results

In this section, we present the results of applying the TT-rules framework to seven datasets, which allow us to demonstrate the effectiveness of our approach and provide evidence for the three claims stated in the introduction.

5.1 Experimental set-up

Evaluation measures and training conditions. We used RMSE, AUC, and accuracy for the evaluation of the regression, binary classification, and multi-class classification respectively. Rules and complexity are defined in Section 3.1.3. All results are presented after grid search and 5-fold cross-validation. All the training features are detailed in the supplementary Section. We compare the performance of our method with that of several other algorithms, including Linear/Logistic Regression [33], Decision Trees (DT)[33], Generalized Linear Models (GL)[44], Neural Additive Models (NAM)[3], XGBoost[16], and Deep Neural Networks (DNNs) [33]. The supplementary materials provide details on the training conditions used for these competing methods. Experiments are available on demand. Our workstation consists of eight cores Intel(R) Core(TM) i7-8650U CPU clocked at 1.90 GHz, 16 GB RAM.

Table 3: Accuracy and complexity on the Compas, Adult and HELOC datasets for different methods. All the TT-rules are computed with our automatic post-training optimizations as described in Section 4.2. TT-rules big refers to a TTnet trained with a final linear regression with weights as floating points, whereas TT-rules small refers to a TTnet trained with a sparse binary linear regression.

	Compas			Adult			HELOC		
	Accuracy	Rules	Complexity	Accuracy	Rules	Complexity	Accuracy	Rules	Complexity
GL	0.685 ± 0.012	16 ± 2	20 ± 6	0.852 ± 0.001	16 ± 1	23 ± 1	0.732 ± 0.001	104 ± 5	104 ± 5
RIPPER	0.560 ± 0.006	12 ± 2	576 ± 48	0.833 ± 0.009	43 ± 15	14154 ± 4937	0.691 ± 0.019	17 ± 4	792 ± 186
DT	0.673 ± 0.015	78 ± 1	12090 ± 155	0.837 ± 0.004	398 ± 5	316410 ± 3975	0.709 ± 0.011	70 ± 1	9522 ± 136
ORS	0.670 ± 0.015	11 ± 1	460 ± 42	0.844 ± 0.006	9 ± 3	747 ± 249	0.704 ± 0.012	16 ± 6	1888 ± 708
TT-rules big (Ours)	0.687 ± 0.005	42 ± 3	4893 ± 350	0.851 ± 0.003	288 ± 12	22896 ± 954	0.733 ± 0.010	807 ± 30	103763 ± 3857
TT-rules small (Ours)	0.664 ± 0.013	13 ± 2	155 ± 22	0.842 ± 0.003	130 ± 10	673 ± 145	0.727 ± 0.010	82 ± 30	574 ± 210

Machine learning datasets. We utilized a variety of healthcare and non-healthcare datasets for our study. For multi-classification, we used the Diabetes 130 US-Hospitals dataset[1] from the UCI Machine Learning Repository [21]. For binary classification tasks, we used two single-cell RNA-seq analysis datasets, one for head and neck cancer[2] [34] and another for melanoma [3][41], as well as the TCGA lung cancer dataset[4] [31] for regression. For binary classification tasks, we used the Adult dataset[5] from the UCI Machine Learning Repository [21], the Compas dataset[6] introduced by ProPublica [7], and the HELOC dataset [1]. We also employed the California Housing dataset[7] [32] for the regression task. Supplementary details regarding each of the datasets can be found in the supplementary Section materials.

DNA datasets. Our TT-rules framework's scalability is demonstrated using two DNA datasets, namely the single-cell RNA-seq analysis datasets for head neck, and melanoma cancer [34, 41] for binary classification and the TCGA lung cancer[31] for regression. These datasets contain 23689 and 20530 features, respectively, and are commonly used in real-life machine learning applications [30, 24, 36, 42].

5.2 Performances comparison - Claim A)

5.2.1 AUC/RMSE/Accuracy - Claim A-1) & A-2)

First, Table 2 demonstrates that our method can handle all types of tasks, including regression, binary classification, and multi-class classification. Moreover, it outperforms most of the other interpretable methods (decision tree, RIPPER, linear/log, NAM) in various prediction tasks, except for GL [44], which performs better than our method on the HELOC dataset. It is worth noting that GL does not support multi-class classification. Additionally, our method shows superior performance to more complex models such as XGBoost and DNNs on California Housing and Compas datasets. Therefore, our method can be considered comparable or superior to the current state-of-the-art methods while providing global and exact interpretability, which will be demonstrated in Section 5.4.

5.2.2 Complexity - Claim A-3)

Impact of post-training optimization. The optimizations proposed in Section 4.2 succeeded to reduce the complexity of our model as defined in Section 3.1.3 at a cost of little accuracy loss as seen in Table 4. The complexity went down by a factor of $1.35\times$, $2.22\times$, and $1.47\times$ on the Adult, Compas, and Diabetes datasets respectively. The accuracy went down for the Adult and Diabetes datasets by 0.004 and 0.009 respectively and stayed the same for Compas.

Table 4: Reduction of the complexity of some TT-rules models after applying optimizations from Section 4.2 on Adult [21], Compas [7] and Diabetes [21] datasets.

Models	TT-rules $\mathcal{R}$		TT-rules $\mathcal{R}_{opt}$	
	Acc.	Complexity	Acc.	Complexity
Adult	0.846 ± 0.003	909 ± 212	0.842 ± 0.003	673 ± 145
Compas	0.664 ± 0.013	343 ± 41	0.664 ± 0.013	155 ± 22
Diabetes	0.574 ± 0.008	$22\text{K} \pm 2800$	0.565 ± 0.009	$15\text{k} \pm 2225$

Comparison with rule-based models. Table 3 presents a comparison of various rule-based models, including ours, on the Compas, Adult, and HELOC datasets, in terms of accuracy, number of rules, and complexity. We note that we report accuracy and AUC for binary classification tasks, as RIPPER and ORS do not provide probabilities. We proposed two TT-rules models: our model for high performances, as shown in Table 2, with floating weights, and a small model with sparse binary weights, which is also our most compact model in terms of the number of rules and complexity. Our proposed model outperforms the others in terms of accuracy on the Compas dataset and has similar performances to GL [44] on the Adult and HELOC datasets. Although GL provides a better tradeoff between performance and complexity, we highlight that GL does not support multi-class classification tasks and is not scalable for larger datasets such as DNA datasets, as shown in the next section. We also propose a small model as an alternative to our high-performing model. Our small model achieves accuracy that is 0.023, 0.009, and 0.006 lower than our best model but requires only $3.2\times$, $2.2\times$, and $9.8\times$ fewer rules on the Compas, Adult, and HELOC datasets, respectively. We successfully reduce the complexity of our model by $14.3\times$, $34\times$, and $180\times$ on these three datasets.

5.3 Scalability - Claim B)

Our TT-rules framework demonstrated excellent scalability to real-life datasets with up to 20K features. This result is not surprising, considering the original TTnet paper [9] showed the architecture's ability to scale to ImageNet. Furthermore, our framework's superiority was demonstrated by outperforming other rule-based models that failed to converge to such large datasets (GL [44], RIPPER [17, 18]). NAMs were not trained as we considered investigating the 20K graphs to be barely interpretable. Regarding performance, the TT-rules framework achieved an impressive RMSE of 0.029 on the DNA single-cell regression problem, compared to 0.092 for linear models, 0.028 for DNNs,

[1] https://bit.ly/diabetes_130_uci
[2] https://bit.ly/neck_head_rna
[3] https://bit.ly/melanoma_rna
[4] https://bit.ly/tcga_lung_rna
[5] https://bit.ly/Adult_uci
[6] https://bit.ly/Compas_data
[7] https://bit.ly/california_statlib

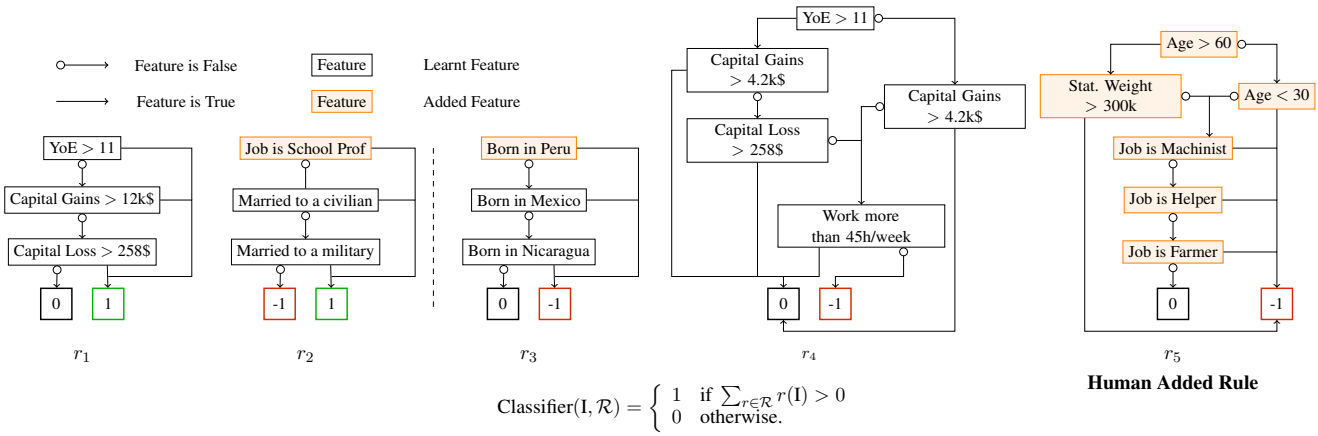


Figure 2: Our neural network model trained on Adult dataset in the form of Boolean decision trees: the output of the DNN and the output of these decision trees are the same, reaching 83.6% accuracy. Added Features are represented in orange rectangles. By modifying existing rules and incorporating r_5, the **Human Added Rule**, we reach 84.6% accuracy. On the same test set, Random Forest reaches 85.1% accuracy and Decision Tree 84.4% with depth 10. There is no contradiction to the rules: one person can not be born in both Mexico and Nicaragua. The term YoE refers to the Years of Education and the Capital Gains (Losses) refer to the amount of capital gained (lost) over the year. Each rule r_i is a function $r_i : \{0,1\}^n \mapsto \{-1,0,1\}$, i.e for each data sample I we associate for each rule r_i a score which is in $\{-1,0,1\}$. The prediction of our classifier is then as stated above.

and 0.42 for Random Forests. On the DNA multi-cross dataset, the TT-rules framework achieved an accuracy of 83.48%, compared to 83.33% for linear models, outperforming DNNs and Random Forests by 10.8% and 10.4%, respectively. Our approach not only scales but also reduces the input feature set, acting as a feature selection method. We generated a set of 1064 rules out of 20530 features for the regression problem, corresponding to a drastic reduction in complexity. For the binary classification dataset, we generated 9472 rules, which more then halved the input size from 23689 to 9472.

5.4 TT-rules application case study - Claim C)

In this section, we present the results of applying the TT-rules framework on the Adult dataset [21], for a specific trained example on Figure 2.

Exact and global interpretability - Claim C-1). For global and exact interpretability, we first apply TT-rules framework to obtain $\mathcal{R}$ and $\mathcal{R}_{opt}$. Then we transform the rules in $\mathcal{R}_{opt}$ into their equivalent ROBDD representation. This transformation is fast and automatic and can be observed in Figure 2: the resulting decision mechanism is small and easily understandable. In the Adult dataset, the goal is to predict whether an individual I will earn more than \$50K per year in 1994. Given an individual's feature inputs I, the first rule of Figure 2 can be read as follows: if I has completed more than 11 years of education, then the rule is satisfied. If not, then the rule is satisfied if I earns more than \$4,200 in investments per month or loses more than \$228. If the rule is satisfied, I earns one positive point. If I has more positive points than negative points, the model predicts that I will earn more than \$50K per year.

Human knowledge injection - Claim C-2). Figure 2 illustrates our model's capability to incorporate human knowledge by allowing the modification of existing rules. However, it is important to note that we do not claim to achieve automatic human knowledge injection. The illustration simply highlights the possibility of manual rule modification in our framework.

Mitigating contextual drift in DNN through global and exact interpretability - Claim C-3). It is essential to recognize that machine learning models may not always generalize well to new data from different geographic locations or contexts, a phenomenon known as "contextual drift" or "concept drift" [23]. The global and exact interpretation of DNNs is vital in this regard, as it allows for human feedback on the model's rules and the potential for these rules to be influenced by contextual drift. For example, as depicted in Figure 2, this accurate model trained on US data is highly biased towards the US and is likely to perform poorly if applied in South America due to rule number 3. This highlights once again the significance of having global and exact interpretability of DNNs, as emphasized by recent NIST Artificial Intelligence Risk Management Framework [4].

6 Limitations and future works

Although our TT-rules framework provides a good balance between interpretability and accuracy, we observed that the generalized linear model (GL) offers a better trade-off. Specifically, for approximately the same performance, GL offers significantly less complexity. As such, future work could explore ways to identify feature interactions that work well together, similar to what GL does. Exploring automatic rule addition as an alternative to the human-based approach used in our work could also be a fruitful direction for future research.

Another interesting avenue is to apply TT-rules to time series tasks, where the interpretable rules generated by our model can provide insights into the underlying dynamics of the data. Finally, another promising area for future work would be to propose an agnostic global explainer for any model based on the TT-rules framework.

7 Conclusion

In conclusion, our proposed TT-rules framework provides a new and optimized approach for achieving global and exact interpretability in regression and classification tasks. With its ability to scale on large datasets and its potential for feature reduction, the TT-rules framework appears as a valuable tool towards explainable artificial intelligence.

References

[1] Fico explainable machine learning challenge. https://community.fico.com/s/explainable-machine-learning-challenge?tabset-3158a=2. Last accessed April 2023.

[2] Ami Abutbul, Gal Elidan, Liran Katzir, and Ran El-Yaniv, 'Dnf-net: A neural architecture for tabular data', *arXiv preprint arXiv:2006.06465*, (2020).

[3] Rishabh Agarwal, Nicholas Frosst, Xuezhou Zhang, Rich Caruana, and Geoffrey E Hinton, 'Neural additive models: Interpretable machine learning with neural nets', *arXiv preprint arXiv:2004.13912*, (2020).

[4] NIST AI, 'Artificial intelligence risk management framework (ai rmf 1.0)', (2023).

[5] Sheldon B. Akers, 'Binary decision diagrams', *IEEE Transactions on computers*, **27**(06), 509–516, (1978).

[6] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin, 'Learning certifiably optimal rule lists for categorical data', *arXiv preprint arXiv:1704.01701*, (2017).

[7] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner, 'Machine bias', in *Ethics of Data and Analytics*, 254–264, Auerbach Publications, (2016).

[8] Seyed Amir Hossein Aqajari, Emad Kasaeyan Naeini, Milad Asgari Mehrabadi, Sina Labbaf, Nikil Dutt, and Amir M Rahmani, 'pyeda: An open-source python toolkit for pre-processing and feature extraction of electrodermal activity', *Procedia Computer Science*, **184**, 99–106, (2021).

[9] Adrien Benamira, Thomas Peyrin, and Bryan Hooi Kuen-Yew, 'Truth-table net: A new convolutional architecture encodable by design into sat formulas', in *Computer Vision – ECCV 2022 Workshops*, eds., Leonid Karlinsky, Tomer Michaeli, and Ko Nishino, pp. 483–500, Cham, (2023). Springer Nature Switzerland.

[10] Christian Bessiere, Emmanuel Hebrard, and Barry O'Sullivan, 'Minimising decision tree size as combinatorial optimisation', in *International Conference on Principles and Practice of Constraint Programming*, pp. 173–187. Springer, (2009).

[11] Armin Biere, Marijn Heule, and Hans van Maaren, *Handbook of satisfiability*, volume 185, IOS press, 2009.

[12] Archie Blake, 'Corrections to Canonical expressions in Boolean algebra', *Journal of Symbolic Logic*, **3**(2), 112–113, (1938).

[13] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci, 'Deep neural networks and tabular data: A survey', *IEEE Transactions on Neural Networks and Learning Systems*, (2022).

[14] Wieland Brendel and Matthias Bethge, 'Approximating cnns with bag-of-local-features models works surprisingly well on imagenet', *arXiv preprint arXiv:1904.00760*, (2019).

[15] Randal E Bryant, 'Graph-based algorithms for boolean function manipulation', *Computers, IEEE Transactions on*, **100**(8), 677–691, (1986).

[16] Tianqi Chen and Carlos Guestrin, 'XGBoost', in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, (aug 2016).

[17] William W Cohen, 'Fast effective rule induction', in *Machine learning proceedings 1995*, 115–123, Elsevier, (1995).

[18] William W Cohen and Yoram Singer, 'A simple, fast, and effective rule learner', *AAAI/IAAI*, **99**(335-342), 3, (1999).

[19] European Commission, 'Proposal for a regulation laying down harmonised rules on artificial intelligence', (2021).

[20] Sanjeeb Dash, Oktay Gunluk, and Dennis Wei, 'Boolean decision rules via column generation', *Advances in neural information processing systems*, **31**, (2018).

[21] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[22] Alex A Freitas, 'Comprehensible classification models: a position paper', *ACM SIGKDD explorations newsletter*, **15**(1), 1–10, (2014).

[23] Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues, 'Learning with drift detection', in *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17*, pp. 286–295. Springer, (2004).

[24] Jasleen K Grewal, Basile Tessier-Cloutier, Martin Jones, Sitanshu Gakkhar, Yussanne Ma, Richard Moore, Andrew J Mungall, Yongjun Zhao, Michael D Taylor, Karen Gelmon, et al., 'Application of a neural network whole transcriptome–based pan-cancer method for diagnosis of primary and metastatic cancers', *JAMA network open*, **2**(4), e192597–e192597, (2019).

[25] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, 'Binarized neural networks', *Advances in neural information processing systems*, **29**, (2016).

[26] Kai Jia and Martin Rinard, 'Efficient Exact Verification of Binarized Neural Networks', in *Advances in Neural Information Processing Systems*, eds., H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, volume 33, pp. 1782–1795. Curran Associates, Inc., (2020).

[27] Arlind Kadra, Marius Lindauer, Frank Hutter, and Josif Grabocka, 'Well-tuned simple nets excel on tabular datasets', *Advances in neural information processing systems*, **34**, 23928–23941, (2021).

[28] Himabindu Lakkaraju, Stephen H Bach, and Jure Leskovec, 'Interpretable decision sets: A joint framework for description and prediction', in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1675–1684, (2016).

[29] Chang-Yeong Lee, 'Representation of switching circuits by binary-decision programs', *The Bell System Technical Journal*, **38**(4), 985–999, (1959).

[30] Yuanyuan Li, Kai Kang, Juno M Krahn, Nicole Croutwater, Kevin Lee, David M Umbach, and Leping Li, 'A comprehensive genomic pan-cancer classification using the cancer genome atlas gene expression data', *BMC genomics*, **18**, 1–13, (2017).

[31] Jianfang Liu, Tara Lichtenberg, Katherine A Hoadley, Laila M Poisson, Alexander J Lazar, Andrew D Cherniack, Albert J Kovatich, Christopher C Benz, Douglas A Levine, Adrian V Lee, et al., 'An integrated tcga pan-cancer clinical data resource to drive high-quality survival outcome analytics', *Cell*, **173**(2), 400–416, (2018).

[32] R Kelley Pace and Ronald Barry, 'Sparse spatial autoregressions', *Statistics & Probability Letters*, **33**(3), 291–297, (1997).

[33] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al., 'Scikit-learn: Machine learning in python', *the Journal of machine Learning research*, **12**, 2825–2830, (2011).

[34] Siddhartha V Puram, Itay Tirosh, Akash S Parikh, Anoop P Patel, and et al., 'Single-cell transcriptomic analysis of primary and metastatic tumor ecosystems in head and neck cancer', *Cell*, **171**(7), 1611–1624.e24, (Dec 2017).

[35] J Ross Quinlan, *C4. 5: programs for machine learning*, Elsevier, 2014.

[36] Egor Revkov, Tanmay Kulshrestha, Ken Wing-Kin Sung, and Anders Jacobsen Skanderup, 'Puree: accurate pan-cancer tumor purity estimation from gene expression data', *Communications Biology*, **6**(1), 394, (2023).

[37] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, '" why should i trust you?" explaining the predictions of any classifier', in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, (2016).

[38] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, 'Anchors: High-precision model-agnostic explanations', in *Proceedings of the AAAI conference on artificial intelligence*, volume 32, (2018).

[39] Ronald L Rivest, 'Learning decision lists', *Machine learning*, **2**(3), 229–246, (1987).

[40] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju, 'Fooling lime and shap: Adversarial attacks on post hoc explanation methods', in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 180–186, (2020).

[41] Itay Tirosh, Benjamin Izar, Sanjay M Prakadan, Mark H 2nd Wadsworth, and et al., 'Dissecting the multicellular ecosystem of metastatic melanoma by single-cell rna-seq', *Science*, **352**(6282), 189–196, (Apr 2016).

[42] Duc Tran, Hung Nguyen, Bang Tran, Carlo La Vecchia, Hung N Luu, and Tin Nguyen, 'Fast and precise single-cell data analysis using a hierarchical autoencoder', *Nature communications*, **12**(1), 1029, (2021).

[43] Zhuo Wang, Wei Zhang, Ning Liu, and Jianyong Wang, 'Scalable rule-based representation learning for interpretable classification', *Advances in Neural Information Processing Systems*, **34**, (2021).

[44] Dennis Wei, Sanjeeb Dash, Tian Gao, and Oktay Gunluk, 'Generalized linear rule models', in *International Conference on Machine Learning*, pp. 6687–6696. PMLR, (2019).

[45] Fan Yang, Kai He, Linxiao Yang, Hongxia Du, Jingbang Yang, Bo Yang, and Liang Sun, 'Learning interpretable decision rule sets: A submodular optimization approach', *Advances in Neural Information Processing Systems*, **34**, (2021).