A New Class of Intelligible Models for Tabular Learning

Kodjo Mawuena Amekoe[1,3], Hanane Azzag[1], Mustapha Lebbah[1,2], Zaineb Chelly Dagdia[2], and Gregoire Jaffre[3]

[1] Sorbonne Paris Nord University - LIPN, UMR CNRS 7030, Villetaneuse, France
[2] Paris-Saclay University - DAVID Lab, UVSQ, Versailles, France
[3] Groupe BPCE, 7 promenade Germaine Sablon 75013 Paris, France
amekoe@lipn.univ-paris13.fr

Abstract. Apart from the high accuracy of machine learning models, what interests many researchers and practitioners in real-life problems (e.g., fraud detection, credit scoring) is finding hidden patterns in the data. In this concern, intrinsically interpretable models are often preferred to complex ones, which are in most cases black-box models. Also, glass-box models such as linear regression or shallow decision trees are used in some high-risk fields to handle tabular data, even if performance must be sacrificed. In this paper, we propose TabSRAs, a new class of accurate tabular learning models with inherent intelligibility. TabSRAs are based on SRA (Self-Reinforcement Attention), a new attention mechanism that helps to learn an intelligible representation of the raw input data through element-wise vector multiplication. The learned representation is aggregated by a highly transparent function (e.g., linear) that produces the final output. Our results on synthetic and real-world data show that TabSRAs perform comparably to state-of-the-art models, while remaining self-explainable. Source code is available at https://github.com/anselmeamekoe/TabSRA.

Keywords: Attention · Interpretability · Tabular Data.

1 Introduction

Since the promising result of the Transformer architecture on machine translation tasks [31], deep learning models continue to provide impressive performance, for example for language modeling or computer vision. Convinced by this utility of the attention mechanism (used in the Transformer architecture), especially in modeling contextual informations, many efforts have been made in using it in order to match or compete the accuracy of boosted tree models such as XGBoost [11] in tabular modeling [28,18,15,13]. The main advantage of these attention based architectures is that they are differentiable, making them easy to use for example in multimodal settings (e.g., encode tabular information with text, image, etc.) or multitask settings [2]. However, these full-complexity models listed above typically use a large number of parameters (or trees), making direct human inspections difficult. On the other hand, interpretability is usually (i) required by regulators in real-world applications (e.g., GDPR: Article 22 in Europe), (ii) desired if the goal is to discover hidden patterns in the data (e.g., fraud detection) or to ensure that the model does not learn a bias that may lead to significant drift in production. Therefore, recent research such as [22,25,21] has focused on developing post-hoc methods

to explain, at least locally, the predictions of full-complexity models. Unfortunately, although these methods provide some interesting properties, they are sometimes based on some computational mechanisms (e.g., exact Shapley value computation) or hypotheses (e.g., independence between features) that are difficult to achieve in practice, leading to biased explanations [6,19]. Still discussing interpretability, [27] provides a technical reason why an interpretable model might exist among the set of accurate models in any domain and encourages researchers to move toward finding this solution, especially for high-risk domains. Convinced by this philosophy, we propose Self-Reinforcement Attention for tabular data (TabSRAs). To achieve the inherent intelligibility of TabSRAs, we found it necessary to develop a representation learning block or layer that (i) preserves the initial feature space (i.e., $\mathbb{R}^p$! $\mathbb{R}^p$), and (ii) reduces to the maximum some extra steps (e.g., residual connection, LayerNorm) that make the overall architecture less interpretable.
Our contributions are summarized as follows:

1. TabSRAs are attention-based supervised models that provide an intrinsic explanation of their predictions and are trained in an end-to-end manner using backpropagation
2. They contain a Self-Reinforcement Attention (SRA) block that is used to learn a *Reinforced* representation of the raw input through element-wise multiplication with the produced attention vector. This consideration allows to: (i) take into account possible interactions without unnecessarily adding additional features (terms) or imposing a limit on the order of interactions between features; (ii) a local and global model understanding, especially using visualization.
3. Our experiments show that our proposed solution provides understandable representations, robust and faithful feature attribution while being competitive in terms of accuracy compared to state-of-the-art models.

The rest of the paper is organized as follows: Section 2 describes the proposed model, and its architecture. The experimental setup, the discussion of the obtained results and the limitations are presented in Section 3. Section 4 presents a brief discussion of related works. Finally, Section 5 concludes the paper and highlights some perspectives.

2 Model Design

The challenge in most supervised tabular learning problems using attention mechanism [28,18,15,13] is to estimate the output $\hat{y} = f_\theta(\mathbf{x})$ given the feature vector $\mathbf{x} = (x_1, ..., x_p)$ 2 $\mathbb{R}^p$. The parametric model f_θ is learned using the training data D = $f(\mathbf{x}_i, y_i)g_{i=1}^n$ with y_i 2 $f0, 1g$ for binary classification or y_i 2 $\mathbb{R}$ for regression tasks.
Our proposed TabSRAs (Fig 1) contain a SRA block (Fig 2) which is a novel attention mechanism layer, denoted as a function $a(.)$.
Given the raw input $\mathbf{x}$, the SRA block produces an attention vector $\mathbf{a} = (a_1, ..., a_i, ...a_p)$. Thereafter the attention vector is used to produce a reinforced input $\mathbf{o} = (o_1, ..., o_i, ..., o_p)$ as follows:

$$\mathbf{o} = \mathbf{a} \quad \mathbf{x} \tag{1}$$

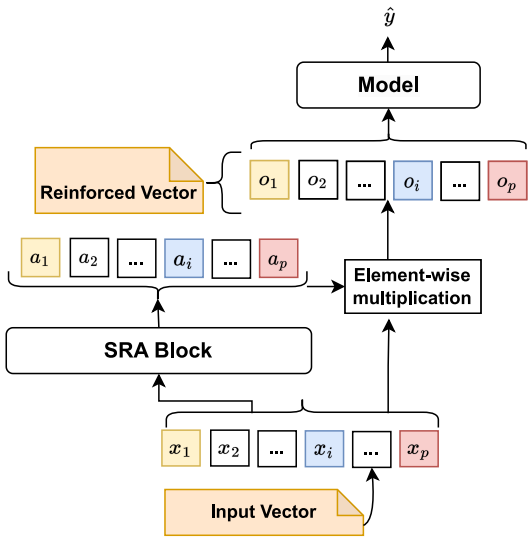


Fig. 1: TabSRAs architecture. The attention vector $\mathbf{a} = (a_1, ..., a_p)$ 2 $\mathbb{R}^p$ provided by the SRA block is used to produce a *reinforced* vector $\mathbf{o} = (o_1, ..., o_p)$ 2 $\mathbb{R}^p$.

where is the element-wise multiplication.
The learned reinforced vector $\mathbf{o}$ represents a new feature basis where each component is guided by the raw input, that is, $o_i = a_i x_i$, helping to maintain the semantics of each dimension. In this space, some components may be shrinked for instance to 0 using the attention weights a_i 0 (see the illustrative example in Fig 3). Indeed, considering the potential internal conflict (due to the interactions) between the raw input components when using linear models for example, the attention weights vector $\mathbf{a}$ may enhance or reduce some components (of the input vector) at strategic and specific positions depending on the context (or the whole information in x). Once interactions managed, it becomes easy to use a highly transparent downstream model to produce the final output.

We investigate in this paper a linear combination of the reinforced feature resulting in the additive TabSRA model (Fig 1). This instantiation called TabSRALinear can be formalized as follows:

$$\begin{aligned} g(\hat{y}) &= \boldsymbol{\beta} \quad \mathbf{o} \\ &= \beta_1 o_1 + ... + \beta_i o_i + ... + \beta_p o_p \\ &= \beta_1 a_1 x_1 + ... + \beta_i a_i x_i + ... + \beta_p a_p x_p \end{aligned} \tag{2}$$

g represents the link function (e.g., usually $g(\mu) = \log(\frac{\mu}{1}{\mu})$ for binary classification and $g = Identity$ for regression tasks).
$\beta_i a_i x_i$ represents the contribution (the prediction importance) of the feature x_i to the output, and $\boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_p)$ is the linear regression coefficient vector.
We argue that the principal added value of TabSRALinear as an intelligible model results in modeling data or phenomena that exhibit feature interactions; otherwise, a simple classical linear model with careful feature engineering should be sufficient to achieve high accuracy.

2.1 SRA block

Given the input vector $\mathbf{x} = (x_1, ...x_i, ..., x_p)$ 2 $\mathbb{R}^p$, the SRA block encodes it into p keys in $K = [\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_i, ..., \mathbf{k}_p]^T$ with $\mathbf{k}_i = (k_i^1, ..., k_i^{d_k})$ 2 $\mathbb{R}^{d_k}$ using a key encoder and queries matrix $Q = [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_i, ..., \mathbf{q}_p]^T$ with $\mathbf{q}_i = (q_i^1, ..., q_i^{d_k})$ 2 $\mathbb{R}^{d_k}$ using a query encoder (Fig.2). The matrix of queries (Q) and keys (K) are generated by

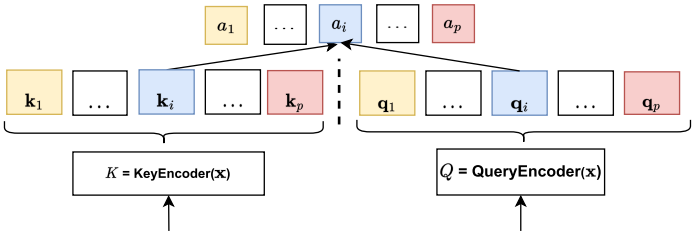


Fig. 2: SRA Block. The $KeyEncoder$ (resp. $QueryEncoder$) produces directly p keys (resp. queries)

two separate fully connected feed-forward networks (FFN) namely $QueryEncoder$ and $KeyEncoder$.

The $KeyEncoder$ (resp. $QueryEncoder$) produces directly p keys (resp. queries) using a single FFN instead of using p independent FFNs per feature, as in [28,15]. This embedding should be particularly useful for heterogeneous (tabular) data especially in the presence of strong features' interactions and at the same time alleviate the need of using several attention blocks (layers) or extra processing which could affect interpretable aspects of the attention coefficients. Furthermore, with a Sigmoid activation function, all elements k_i^j of K (resp. q_i^j of Q) are scalar numbers bounded in $[0, 1]$.

The keys in K are compared to the queries Q component by component using the scalar product as in [31]. This allow to quantify the alignment of different transformations of the same input calculating the attention weights $\mathbf{a} = (a_1, .., a_i, ..., a_p)$ as follows :

$$a_i = \frac{\mathbf{q}_i \quad \mathbf{k}_i}{d_k} \quad \text{for} \quad i \; 2 \; 1, \qquad , p \tag{3}$$

We further use the scaling by d_k in order to reduce the magnitude of the dot-product and to obtain dimension-free attention coefficients a_i 2 $[0, 1]$.
We propose this attention computation in order to learn a reinforced representation that preserves the local proximity of the data points (please refer to Section 2.2 for more details). Moreover we found it efficient in producing a concise explanation of the decision process by providing important number of almost zero attention weights depending on the problem.

2.2 Why SRA block for Robust Representation Learning

In this section, we provide a theoretical analysis regarding the stability of representations learned with the SRA block and justify why TabSRAs are a viable solution for

robust self-explainability in tabular learning settings.
The robustness of representation learning algorithms is usually desired in order to preserve, for example, the proximity among data points with respect to the initial space. Regarding TabSRAs, the robustness can be assessed using the reinforced representations or feature attributions (for instance, TabSRALinear). For feature attribution based explanations system, the robustness remains an important topic with the goal of designing the convenient metric to assess the similarity of explanations provide for similar inputs [4,5,1]. This is mainly because many state-of-the-art interpretability tools [22,25,21] operates on single datapoint, and use point-wise explanation to understand complex model is perhaps too optimistic or can lead to false sense of understanding [5]. To address this limitation, one might want to go beyond individual points and examine the behavior of the models in the neighborhood of some target points. Therefore it's crucial for interpretability methods or self-explainable models to produce explanations that are robust to local perturbations [5].
The TabSRALinear model is designed to produce relatively robust explanations considering the following theorem:

Theorem 1. *The reinforced representation learned using the SRA block (Equation 1) and the feature attribution produced by TabSRALinear (Equation 2) are locally stable in the sense of Lipschitz, that is, for all* $\mathbf{x}$ *2* $\mathbb{R}^p$*, there exists* $\delta > 0$ *and* $L_x \quad 0$ *finite such that:*

$$k\mathbf{x} \quad \mathbf{x}^0 k_1 < \delta \; =) \quad k\beta \quad a(\mathbf{x}) \quad \mathbf{x} \quad \beta \quad a(\mathbf{x}^0) \quad \mathbf{x}^0 k_1 \quad L_x k\mathbf{x} \quad \mathbf{x}^0 k_1 \qquad (4)$$

With $L_x = k\beta k_1 \, (k\mathbf{a} k_1 + L_a k\mathbf{x} k_1)$ *and* $L_a > 0$ *the Lipschitz constant of the SRA block.*

The objective of the Theorem 1 is not to provide the tightest bound of the Lipschitz constant but to provide some justification for attention computation.
First of all, we can notice the quantity $k\mathbf{x}k_1$ in the expression of L_x, which shows that raw input data should be bounded. This is common situation when using deep learning models. That is, data scaling technique (using the minimum and the maximum or the mean and the standard deviation) or quantile transformation is used to speed up the convergence. We also have the term $k\mathbf{a}k_1$ which proves the smaller are the attention weights the more stable are the explanations. Using the scaling of Equation 3 we have $k\mathbf{a}k_1 = 1$ and we can identify in the first term of L_x the Lipschitz constant of linear models, which is simply $k\beta k_1$. Moreover in situations where there is no interactions (and non-linear effects) between features, almost all attention weights are expected constant (i.e., $L_a \quad 0$), and TabSRALinear is reduced to classical linear models.
Owing to space limitations, we do not provide a complete proof of the Theorem 1. However this can be easily done using the fact that (i) a fully connected layer (linear transformations followed by common 1-Lipschitz activation such as ReLU or Sigmoid) is Lipschtiz continuous [17] (ii) The product of two Lipschitz continuous and bounded function is Lipschitz continuous.

We demonstrate empirically (Section 3.2) that TabSRALinear's feature attributions are more robust and faithfull than using state-of-the-arts interpretability tools such as [21,22].

Table 1: Some differences between the SRA block and the classical Transformer block.

Point	Classical Transformer block	SRA block
Attention weight	$A = softmax(\frac{QK^T}{\sqrt{d_k}}) \in \mathbb{R}^{p \times p}$	$\mathbf{a} = \frac{\sum_{d_k} Q \odot K}{d_k} \in \mathbb{R}^p$
Value encoding	Yes	No
Additional processing (residual connection, LayerNorm)	Yes	No

2.3 Difference between SRA and others Transformer block

In this section we provide some important differences between the proposed SRA block and the Transformer block used in [28,18,15,13,31]. We denote by $K \in \mathbb{R}^{p \times d_k}$ the matrix of keys and $Q \in \mathbb{R}^{p \times d_k}$ the matrix of queries and summarize these differences in Table 1.

3 Experiments

3.1 Experimental setup

Our motivation when building the TabSRAs is to combine interpretability and performance in a single model. Typically, the interpretability of models is assessed separately from their performance, which can make it challenging to gauge the effectiveness of our solution. Nonetheless, we believe it is appropriate to measure the value of our proposition by comparing it to both glass-box and full-complexity benchmarks using the following criteria:

- **Intelligibility**: Are the representations learned using the SRA block understandable? Are the explanations provided by TabSRALinear faithful? Are the explanations concise enough to be understood by humans? Are the explanations robust/coherent, i.e., similar for similar inputs?
- **Effectiveness**: Is TabSRALinear accurate compared to state-of-the-art models?

Model setup.

- Choice of the query and key encoder: we use the same architecture for the key and query encoders which is a two ReLU activation hidden layers fully connected neural network of dimension $[d_1, d_2]$ with, $d_1 = p \quad (d_k/4)$ and $d_2 = p \quad (d_k/2)$, $d_k \quad 4$.
- Regularization: to increase the generalization power, we used regularizations in the SRA block. Specifically, we used dropout [29] in both the key and query encoders during the training. Also, we used weight decay (L_2 penalization) to empower the smoothness in the embeddings (of the key and query).

Table 2: Benchmark datasets

Datasets	# Datapoints	# features	# Categorical features	Positive Class (%)
Bank Churn	10000	10	2	20.37
Credit Default	30000	23	3	22.16
Bank Marketing	45211	16	9	11.70
Adult Income	30162	14	8	24.89
Credit Card Fraud	284807	29	0	0.17
Blastchar	7043	19	16	26.54
Telco Churn	66469	63	0	20.92
Heloc Fico	10459	23	0	47.81

Benchmark datasets. As we focus particularly on finance as an application domain, we considered three UCI datasets (Default of Credit Card Clients, Bank Marketing, and Adult Income) and four Kaggle datasets (Credit Card Fraud, Bank Churn Modelling, Blastchar and Telco Churn), and the Heloc Fico dataset for our experiments. All of these datasets are used for binary classification tasks, and the number of features (both numerical and categorical) ranges from 10 to 63. The percentage of positive class instances varies between 0.17% and 48% (see Table 2 for further details). Unless otherwise specified, all categorical inputs are one-hot encoded, and numerical inputs are scaled using mean and standard deviation to accelerate the convergence of the algorithms.

Benchmark models. We compare quantitatively the accuracy/intelligibility of TabSRALinear (Equation 2) with the following baselines:

- **Logistic/Linear Regression (LR)**: It is a highly interpretable model obtained by simple linear combination of features followed by Sigmoid activation for binary classification problem.
- **Decision Trees (DT)**: It is another glass-box model that provides decision rules following the paths in the obtained tree. We used the scikit-learn implementation[9].
- **TabNet** [7]: Uses a multiplicative sparse mask vector for instance-wise feature selection and the masks are aggregated across many stages to compute features' importance and a local explanation of its predictions.
- **MultiLayer Perceptron (MLP)**: it is a full complexity architecture that can model nonlinear effects and interactions. It somehow provides us the achievable accuracy by shallow and differentiable architectures. We consider two hidden layers MLP (with ReLU activation) of dimensions $f4 \quad p, 2 \quad pg$ as in [15]. p is the input feature dimension.
- **XGBoost** [11]: Remains the leading state-of-the-art model for several real-life use cases, and tabular learning competitions. We compare the intelligibility of TabSRALinear with XGBoost coupled with TreeSHAP [21]. TreeSHAP is a well-suited and computationally efficient explanation tool for tree based models.

It is also to be noted that we do not compare directy to some attention-based models, such as [28,18,15,13] as they are more motivated by performance than interpretability and XGBoost can give an idea of the upper bound that these models can reach in most cases [14,8,13].

Accuracy evaluation measures. We evaluate the models on benchmark datasets using 5-stratified fold cross-validation (80% for the training) and report the mean and standard deviation of the Area Under the ROC curve (AUCROC) on the validation set. Particularly for highly imbalanced datasets (e.g., the Credit Card Fraud dataset), we optimize and reported the Average Precision or Precision-Recall (AUCPR). In fact, AUCPR gives a more informative picture of an algorithm's performance than AUCROC in highly imbalanced data settings and algorithms that optimize AUCPR are guaranteed to optimize AUCROC [12].

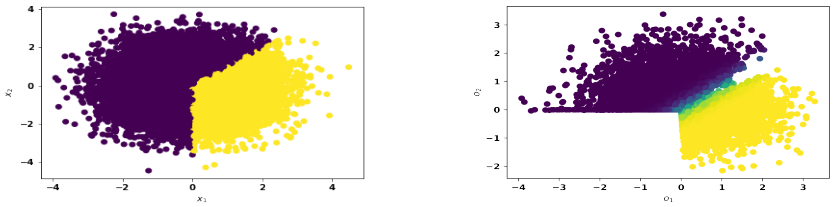
Training details. For TabSRALinear, MLP and LR we used the Pytorch library [23] and for XGBoost we used [11]. For each benchmark dataset and model, the hyperparameters are optimized on the validation set of the first cross validation split using 30 trials of Bayesian optimization (using Optuna [3]). These hyperparameters are then used for the 4 remaining splits and the best checkpoints performance are reported for all datasets and models following [10]. In this way, we avoid computational overheads and at the same time we favor models that are less sensitive to the hyperparameters. We also set the maximum hyperparameters searching time to 20 hours (per fold and model). Only TabNet has used the entire 20 hours for the Credit Card Fraud dataset.

3.2 Intelligibility of TabSRALinear

How raw data are reinforced using the SRA block. To illustrate how the raw data is reinforced in practice, we use 2D toy datasets with the objective of facilitating the visualization. First, we consider the following function:

$$F_1(x) = 5x_1 \quad 5x_2\mathbb{1}_{x_1>0} \quad \text{and} \quad y = f\mathbb{1}_{p>0.5} \quad \text{with} \quad p = 1/(1+e^{\ F_1(x)}g \quad (5)$$

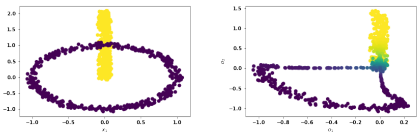
As simple as it may seem, this function cannot be directly modeled with a linear model due to the term $\mathbb{1}_{x_1>0}$, which forces x_2 to have no effect on the output when $x_1 < 0$. Using the reinforced version of the raw inputs helps to alleviate this problem; as shown in Fig 3. Fig 3a shows the original data distribution, with the yellow color indicating the class of interest. In Fig 3b, we show the representation learned by multiplying the raw inputs with SRA coefficients. The green color represents a decision boundary to separate the two classes. Through multiplication, values of x_2 are significantly reduced for instance to 0 when needed (i.e., $o_2 \quad 0$ when $x_1 < 0$, $x_2 < 0$), which makes the classes easy to separate with the downstream model which is a simple linear model in this work. Moreover, this representation helps to understand the global behavior of the TabSRALinear model, where it is confident in predicting class 1 (in yellow color) and where it is less confident, as highlighted by the green color.



(a) $\mathbf{x}$ (b) $\mathbf{o} = \mathbf{a} \quad \mathbf{x}$

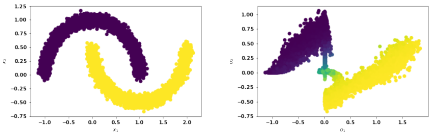
Fig. 3: Illustration of the reinforcement process on 7500 synthetic data points with 0 mean, unit variance Gaussian distribution. The yellow color is used for the class of interest.

We included two more examples, the 2D chainLink [30], the Noisy two moon as depicted in Fig4 and Fig5. By applying SRA coefficients to this dataset, we acquired a new data representation that enables the easy separation of classes, as shown in Fig4b. Even without knowledge of the true data generating process, it is apparent that all observations have been moved strategically so that a simple rule, can effectively isolate nearly all yellow observations of interest.



(a) $\mathbf{x}$ (b) $\mathbf{o} = \mathbf{a} \quad \mathbf{x}$

Fig. 4: ChainLink 2D: 1000 data points



(a) $\mathbf{x}$ (b) $\mathbf{o} = \mathbf{a} \quad \mathbf{x}$

Fig. 5: Noisy two moons: 10000 data points

Can the TabSRALinear identify important features? In order to interpret machine learning models, it is sometimes essential to perform feature attribution, which involves identifying which variables contributed to a high output score.

We aim to assess TabSRALinear's ability to identify crucial features in comparison to that of Linear Regression, TabNet and XGBoost coupled with TreeSHAP [21]. As the ground truths are generally unavailable for real-life datasets, we generate three synthetic datasets with 5 features $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)$ of size 30000 based on the Gaussian distribution (of mean 0 and variance 1) as follows:

$$\textit{Synthetic 1}: \quad y = 5x_1 \quad 5x_2 \tag{6}$$

$$\textit{Synthetic 2}: \quad y = x_1^2 \mathbb{1}_{x_1 \quad 0} \tag{7}$$

$$\textit{Synthetic 3}: \quad y = (5x_1 \quad 5x_2)\mathbb{1}_{x_5 \quad 0} + (5x_3 \quad 5x_4)\mathbb{1}_{x_5 > 0} \tag{8}$$

For these three datasets, we use 80/20% train/test split and we assess each model on the test part.

The example called *Synthetic 1* is linear regression friendly and only x_1 and x_2 are

Table 3: Relevant feature discovery capacity. Precision (%) is used as metric. R^2 (the higher the better) is used to evaluate the test performance (accuracy). The bold numbers denote the best for each dataset and metric.

Datasets	Models	Precision	Test performance
Synthetic 1	LR	**100.00**	**100.00**
	TabNet	65.67	99.97
	TabSRALinear	100.00	100.00
	XGBoost+TreeSHAP	99.85	99.40
Synthetic 2	LR	96.82	50.14
	TabNet	49.97	**99.99**
	TabSRALinear	**99.79**	99.98
	XGBoost+TreeSHAP	99.73	99.30
Synthetic 3	LR	51.28	49.91
	TabNet	54.33	99.20
	TabSRALinear	**99.75**	**99.72**
	XGBoost+TreeSHAP	75.55	99.21

relevant. The goal is to verify whether TabSRALinear can be reduced to simple linear model when required (as discussed in Section 2.2). The example *Synthetic 2* represents a branch of a parabolic function. Linear regression cannot therefore accurately model this example, although it can provide the most important inertia (direction). We consider in the test set observations where $x_1 \quad 0$ (3743 data points) hence only x_1 should have nonzero importance. The *Synthetic 3* example (borrowed from [6]) highlights interactions between the features. Feature interactions is a well-known situations where most of post-hoc explanation tools struggle to find 'truly' relevant features of the underlying model [6,19,16]. For this example, a perfect model should use only the features x_1 and x_2, or alternatively, depending on the sign of x_5, use the features x_3 and x_4. We restrict our analysis to those data points with $x_5 \quad 0$, which comprise 3750 instances. Therefore only x_1 and x_2 are relevant among (x_1, x_2, x_3, x_4). We use the precision in finding the most relevant features as evaluation metric while the R^2 is used to assess the test performance (accuracy).

As shown in Table 3, TabSRALinear is able to accurately detect the most relevant features of *Synthetic 3* with high precision. Moreover the achieve discovery precision matches the one of the best baseline on datasets without features interactions that is LR for the Synthetic 1 and XGBoost+TreeSHAP for Synthetic 2.

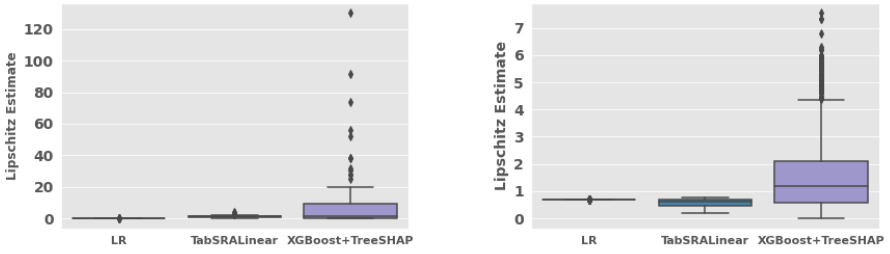
From these synthetic examples, we can identify two possible biases when using feature attribution: (i) the first is due to underfitting (e.g., using linear models to fit complex data); (ii) the second is due to post-hoc interpretabilty tools used to explain full complexity models. In the context mentioned above, the TabSRALinear model appears to be a good compromise for both the feature attribution and accuracy aspects.

Empirical evaluation of the robustness. We evaluate the robustness of TabSRALinear to input perturbation using two real word well-known datasets: Credit Card Fraud and Heloc Fico 2. More specifically we consider the continous notion of stability [5] and

we estimate the local Lipschitz constant as follow:

$$\hat{L}(x) = \underset{x^0 \in N_\epsilon(x)}{\arg\max} \| f_{expl}(\mathbf{x}) \quad f_{expl}(\mathbf{x}^0) \|_2 / \| \mathbf{x} \quad \mathbf{x}^0 \|_2 \tag{9}$$

where the vector $f_{expl}(\mathbf{x})$ is the feature attribution (the explanation) for the given observation x. For Linear models this vector is $\boldsymbol{\beta} \quad \mathbf{x}$ while for TabSRALinear it is $\boldsymbol{\beta} \quad a(\mathbf{x}) \quad \mathbf{x}$. For Linear models, $\hat{L}(x)$ is equivalent to $\| \boldsymbol{\beta} \|_1$ and for TabSRALinear it is proportional to L_x (see Theorem 1). We generate 100 neighbors $N_\epsilon(x)$ by adding a random Gaussian noise $N(0, \epsilon \quad I)$ to every target point x. The used perturbation is small enough to avoid excessively changing the predictions. That is we use $\epsilon = 0.001$ for the Credit Card data and $\epsilon = 0.01$ for the Heloc Fico dataset.
As shown with Fig 6 TabSRALinear's explanations are robust to perturbations compared to the pipeline XGBoost+TreeSHAP while providing almost the same accuracy on these two datasets (see Table 4 for more details). Surprisingly, the Lipschitz estimate of TabSRALinear is nearly close to the one of Linear models (LR) which are known to be robust and conservative. This show that the effect applying the SRA attention weights is similar for very close datapoints.



(a) Credit Card Fraud dataset: 99 test points of class 1

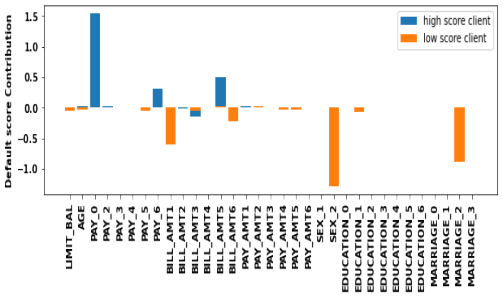
(b) Heloc Fico dataset: 1000 test points of class 1

Fig. 6: Estimation of Lipschitz constant on real word datasets. LR = Logistic Regression

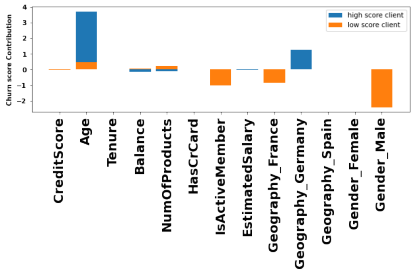
Individual prediction explanation. Another criterion when evaluating model interpretabity is whether the explanations are human-friendly, meaning they are (i) concise enough to be understood by humans (ii) in accordance with the data knowledge or domain experts. We validate our proposed model by considering these two points. For this purpose we used the Credit Default and the Bank Churn Modeling datasets (Table2). In the case of high-risk clients shown in Fig 7a, TabSRALinear focuses mainly on PAY_0, which represents the repayment status in the last month before the prediction. The repayment status is the number of months of delay in the payment, ranging from -2 (i.e., the client has paid two months in advance) to 8 (8 months overdue). Our exploratory analysis confirms that PAY_0 is the most important risque factor for this dataset. For the low risk clients, the model focused more on SEX_2 (female) and MARRIAGE_2. These two categories have slightly low default risk (compared to the other categories

of the same variables), but depending on the other characteristics of this client (e.g., repayment status), TabSRALinear amplifies and uses them to produce low output score.

For Bank Churn Modeling (Fig 7b), according to our data exploration customers with important number of bank products (NumOfProducts$>$ 2) tend to close their accounts and the churn rate the customers in Germany (Geography=Germany) is twice that of other countries. The Pearson's correlation also indicates that the Age is positively correlated to the target. We consider a high churn risk customerwith NumOfProducts=1 (meaning that NumOfProducts is not a risk factor for this customer). The model indicates that Age and Geography=Germany are the two most important features, which confirms our data knowledge. On the other hand, the low churn risk costumer is a active member (IsActiveMember=1) and is a male (the churn rate is 16% for Gender=Male and 25%, remaining).



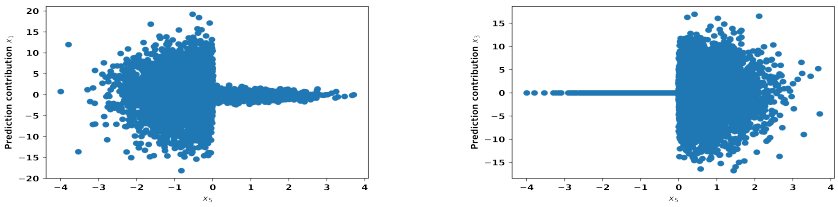
(a) Credit Default: The goal is to predict the default probability (for the next month) of credit card clients in Taiwan, using historical data.



(b) Bank Churn Modeling: the objective is to predict whether the customer will leave the bank (close his account) or continues to be a customer.

Fig. 7: Individual Prediction explanation

Limitations of TabSRAs based explanations. TabSRAs for instance TabSRALinear, as proposed, should not be used directly as a global feature selector but rather after identifying all relevant variables. This is because the feature importance measure provided by TabSRALinear is 'the local' prediction importance and not 'the global' feature importance (cf. Equation 2). Although these two terms are usually used interchangeably in the literature of feature attribution methods, there are some nuances [20]. Specifically, the feature that is important to a local prediction is automatically relevant globally, but the inverse is not always true, particularly when there are interactions. Regarding the TabSRALinear model, an illustrative example is the *synthetic 3* dataset (Equation 8). For this dataset, a perfect TabSRALinear model will almost always give zero prediction importance to the feature x_5 as it cannot be used as main effect or feature (although it can be used to reduce the contribution of other features in the attention vector). Thus, based solely on the prediction importance or its mean aggregation over all test points , one may be tempted to delete feature x_5 in order to create a model with fewer variables. However with further analysis (e.g., visualizing or computing the gradient $\beta_i a_i x_i$ vs x_5) we can notice that x_5 must be maintained. An shown in Fig 8, an important information



(a) Prediction importance of x_1 vs x_5 (b) Prediction importance of x_3 vs x_5

Fig. 8: *synthetic 3*: relevance analysis of the feature x_5

Table 4: Accuracy of the TabSRALinear model. Mean and standard deviation AUC (%), reported from a 5-stratified cross validation. Bold highlights the best performance when comparing self-explainable (LR, DT, TabNet, TabSRALinear) models and italic is used for the overall best performing model.

Datasets	LR	DT	TabNet	TabSRALinear	MLP	XGBoost
BankChurn	76.93 1.56	84.13 0.75	**86.99** 0.79	86.98 0.46	*87.08* 0.73	86.82 0.79
CreditDefault	72.53 0.49	76.26 0.99	**77.85** 1.03	77.55 0.56	78.24 0.78	*78.56* 0.69
BankMarketing	90.79 0.49	90.92 041	92.74 0.70	**93.33** 0.50	93.44 0.41	*93.82* 0.38
AdultIncome	90.50 0.41	90.11 0.59	90.46 0.52	**91.07** 0.42	91.45 0.38	*92.63* 0.37
CreditCardFraud	77.08 2.59	79.66 4.42	81.09 3.92	**86.58** 2.81	85.69 2.53	86.54 2.19
Blastchar	84.54 1.48	83.49 1.21	83.53 1.45	**84.63** 1.51	84.63 1.52	*84.89* 1.21
TelcoChurn	88.95 0.29	90.21 0.34	90.45 0.33	**90.52** 0.31	90.54 0.28	*91.13* 0.37
HelocFico	78.26 0.52	76.76 0.90	79.39 0.57	**79.43** 0.41	79.50 0.46	*79.75* 0.74

needs to be known about x_5; which is its sign. When $x_5 < 0$ (resp. $x_5 > 0$), the prediction contribution (or importance) of x_3 is close to 0 (resp. the prediction contribution of x_1 is close to 0). A similar visualization would lead to the same finding for the contributions of x_2 and x_4, indicating that x_5 is relevant to the model. Dropping it would result in a drastic reduction in TabSRALinear's performance, as it would behave like a simple linear regression.

3.3 The effectiveness of the SRA block.

In this section, we discuss the effectiveness of the SRA block by comparing the accuracy achieved by TabSRALinear model (Equation 2) on benchmark datasets relatively to the baseline models (interpretable and non-counterparts).

As shown in Table 4, TabSRALinear achieved the best performance in 6/8 cases among the self-explainable models (over TabNet, DT and LR). Furthermore, the obtained performance is often close (for 6/8 benckmark datasets) to the one of the overall best performing model which is XGBoost. These results confirms the effectiveness of SRA block particularly when observing the difference of performances between the Logistic Regression (LR) and TabSRALinear which ranges from +0.09 for the Adult-Income dataset to +10.05 AUC for the Bank Churn dataset. We recall that LR model is the resulting architecture when removing the SRA block or setting the attention weights to 1 (cf. Fig 1).

4 Related work

Recently, many deep learning models were designed in the favor of inherent intelligibility. Among these models, we mention Neural Additive Models (NAMs) [2] which presented a neural implementation of the classic Generalized Additive Models (GAMs). In addition to the marginal feature contribution, NAMs provides the shape function which by visualization can help understand the global contribution of a given feature. NODE-G^2AM [10] is an improvement of NAMs built on top of the NODE architecture [24] to take into account pairwise interactions among features. Among the drawbacks of these deep learning architectures is that they apply one sub-network per feature (or pair of features) which can be very resource consuming, especially for high dimensional problems, and the management of higher-order interactions is not guaranteed.
Perhaps the works most closely related to ours are [4,26]. The global idea behind these works is to imitate the formulation of classical linear models while allowing the regression coefficients to vary. Specifically, the final prediction is given by $g(\hat{y}) = \theta(\mathbf{x}) \quad \mathbf{x}$. However, without further condition on $\theta(\mathbf{x})$, such model is no more interpretable than any deep neural networks as $\theta(\mathbf{x})$ may vary significantly [4]. To overcome this problem a linearization approach is used to approximate the behavior of the neural network θ with a linear model in a local region around a given input resulting in SENN (Self-Explaining Neural Network) [4]. However a too high of value the penalization parameter (use for the linearization process) may result in a drastic decrease in the model's performance, and it is not always clear which value of this parameter should be chosen to balance the trade-off between model performance and interpretability, and to determine when the model can be considered as self-explainable.
In constrast to SENN, our proposed solution in this work for instance TabSRALinear didn't required any additional parameter for controlling the linearization although the theorithical analysis (Section 2.2) as well the experiments demonstrate that TabSRALinear can accuretely approx linear functions (Cf. 3.2) and provide robust explanations.

5 Conclusion and Future Work

We presented a new class of intelligible models for tabular learning named TabSRAs and investigated in this work an additive version called TabSRALinear. TabSRAs are based on Self-Reinforcement Attention (SRA), an attention based representation learning block that produces a reinforced version from raw input data through element-wise multiplication. We also showed that TabSRALinear is intelligible in sense that it provides an understandable intermediate representation and an intrinsic feature attribution. Our experimental results confirms the proposed model as a promising solution for self-explainable models in tabular learning settings without the need to 'sacrificing the accuracy'. Overall, we recommend to the interested user to check as much as possible the agreement of the TabSRAs based explanations with their data knowledge since these are not causalities. The SRA block as proposed can be further enriched especially to deal with complex tasks. In this concern, we are currently working on how to use several heads and layers, similar to what is often performed in attention-based architectures. Also, combine the SRA block with rule based models (e.g., decision trees) is an impor-

tant direction of future research as well as incorporating data knowledge in the training phase (e.g., monotonic constraints with respect to some features).

References

1. Agarwal, C., Johnson, N., Pawelczyk, M., Krishna, S., Saxena, E., Zitnik, M., Lakkaraju, H.: Rethinking stability for attribution-based explanations (2022)
2. Agarwal, R., Melnick, L., Frosst, N., Zhang, X., Lengerich, B., Caruana, R., Hinton, G.E.: Neural additive models: Interpretable machine learning with neural nets. Advances in Neural Information Processing Systems **34**, 4699–4711 (2021)
3. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2623–2631 (2019)
4. Alvarez Melis, D., Jaakkola, T.: Towards robust interpretability with self-explaining neural networks. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018)
5. Alvarez-Melis, D., Jaakkola, T.S.: On the robustness of interpretability methods. arXiv preprint arXiv:1806.08049 (2018)
6. Amoukou, S.I., Salaün, T., Brunel, N.: Accurate shapley values for explaining tree-based models. In: International Conference on Artificial Intelligence and Statistics. pp. 2448–2465. PMLR (2022)
7. Arik, S.Ö., Pfister, T.: Tabnet: Attentive interpretable tabular learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 6679–6687 (2021)
8. Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., Kasneci, G.: Deep neural networks and tabular data: A survey. arXiv preprint arXiv:2110.01889 (2021)
9. Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., et al.: Api design for machine learning software: experiences from the scikit-learn project. arXiv preprint arXiv:1309.0238 (2013)
10. Chang, C.H., Caruana, R., Goldenberg, A.: Node-gam: Neural generalized additive model for interpretable deep learning. arXiv preprint arXiv:2106.01613 (2021)
11. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785–794 (2016)
12. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proceedings of the 23rd international conference on Machine learning. pp. 233–240 (2006)
13. Gorishniy, Y., Rubachev, I., Khrulkov, V., Babenko, A.: Revisiting deep learning models for tabular data. Advances in Neural Information Processing Systems **34**, 18932–18943 (2021)
14. Grinsztajn, L., Oyallon, E., Varoquaux, G.: Why do tree-based models still outperform deep learning on tabular data? arXiv preprint arXiv:2207.08815 (2022)
15. Huang, X., Khetan, A., Cvitkovic, M., Karnin, Z.: Tabtransformer: Tabular data modeling using contextual embeddings. arXiv preprint arXiv:2012.06678 (2020)
16. Huang, X., Marques-Silva, J.: The inadequacy of shapley values for explainability. arXiv preprint arXiv:2302.08160 (2023)
17. Kim, H., Papamakarios, G., Mnih, A.: The lipschitz constant of self-attention. In: International Conference on Machine Learning. pp. 5562–5571. PMLR (2021)
18. Kossen, J., Band, N., Lyle, C., Gomez, A.N., Rainforth, T., Gal, Y.: Self-attention between datapoints: Going beyond individual input-output pairs in deep learning. Advances in Neural Information Processing Systems **34**, 28742–28756 (2021)

19. Kumar, I.E., Venkatasubramanian, S., Scheidegger, C., Friedler, S.: Problems with shapley-value-based explanations as feature importance measures. In: International Conference on Machine Learning. pp. 5491–5500. PMLR (2020)
20. Lemhadri, I., Li, H.H., Hastie, T.: Rbx: Region-based explanations of prediction models. arXiv preprint arXiv:2210.08721 (2022)
21. Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.I.: From local explanations to global understanding with explainable ai for trees. Nature machine intelligence **2**(1), 56–67 (2020)
22. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Advances in neural information processing systems **30** (2017)
23. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
24. Popov, S., Morozov, S., Babenko, A.: Neural oblivious decision ensembles for deep learning on tabular data. arXiv preprint arXiv:1909.06312 (2019)
25. Ribeiro, M.T., Singh, S., Guestrin, C.: " 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)
26. Richman, R., Wüthrich, M.V.: Localglmnet: interpretable deep learning for tabular data. Scandinavian Actuarial Journal pp. 1–25 (2022)
27. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature machine intelligence **1**(5), 206–215 (2019)
28. Somepalli, G., Goldblum, M., Schwarzschild, A., Bruss, C.B., Goldstein, T.: Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. arXiv preprint arXiv:2106.01342 (2021)
29. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
30. Ultsch, A.: Clustering wih som: U* c. Proc. Workshop on Self-Organizing Maps (01 2005)
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)