Front. Comput. Sci., 2025, 19(5): 195316
https://doi.org/10.1007/s11704-024-40117-2

RESEARCH ARTICLE

Interpretation with baseline shapley value for feature groups on tree models

Fan XU, Zhi-Jian ZHOU, Jie NI, Wei GAO (✉)

National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
School of Artificial Intelligence, Nanjing University, Nanjing 210023, China

Abstract Tree models have made an impressive progress during the past years, while an important problem is to understand how these models predict, in particular for critical applications such as finance and medicine. For this issue, most previous works measured the importance of individual features. In this work, we consider the interpretation of feature groups, which is more effective to capture intrinsic structures and correlations of multiple features. We propose the *Baseline Group Shapley value* (short for BGShapvalue) to calculate the importance of a feature group for tree models. We further develop a polynomial algorithm, *BGShapTree*, to deal with the sum of exponential terms in the BGShapvalue. The basic idea is to decompose the BGShapvalue into leaves' weights and exploit the relationships between features and leaves. Based on this idea, we could greedily search salient feature groups with large BGShapvalues. Extensive experiments have validated the effectiveness of our approach, in comparison with state-of-the-art methods on the interpretation of tree models.

Keywords interpretability, shapley value, random forests, decision tree

1 Introduction

Past years have witnessed impressive successes for tree models such as random forests [1], XGBoost [2], and deep forests [3], while it is always accompanied by an important problem of understanding how these models make predictions. Reliable interpretations could uncover models' essence, and explain their performance; this is particularly necessary for some critical applications such as finance, medicine and autonomous driving [4–8].

Various methods have been proposed to measure feature importance for the interpretability of tree models. Breiman et al. [9] studied feature importance based on the impurity reduction during the process of splitting. Strobl et al. [10] calculated feature importance by the splitting number of a feature. Louppe et al. [11] studied impurity-based feature importance for tree ensembles in asymptotic sample conditions. Saabas [12] computed feature importance by the changes of average outputs along the decision path. Kazemitabar et al. [13] established performance guarantees for impurity-based feature importance with finite samples. Li et al. [14] also analyzed the feature selection bias of impurity-based methods.

The Shapley value [15] has been applied to interpret tree models in recent years. For example, Lundberg et al. [16] applied the Shapley value to present consistent interpretations for tree models. Some studies exploited Shapley values to explain XGBoost for various applications [17,18]. Sutera et al. [19] studied the relationships between Shapley values and impurity reduction. Amoukou et al. [20] proposed a more accurate calculation of Shapley values for tree models.

Most studies concentrate on the interpretations of individual features for tree models regardless of plentiful correlations among features. In practice, it is natural to present some interpretations over feature groups. Intuitively speaking, the latter is more effective to capture intrinsic structures and correlations among multiple features, and this is beneficial to understanding models' predictions.

In this work, we study baseline Shapley value [21] for the importance of feature groups on tree models, which presents intuitive interpretations and is also flexible to model the interpretation context w.r.t. different applications. To our knowledge, this is the first extension of baseline Shapley value to feature groups, and the main contributions of this work can be summarized as follows:

- Motivated from game theory, we propose the *Baseline Group Shapley value* (BGShapvalue) to quantify the importance of a feature group, together with some desirable properties. The basic idea is to utilize an auxiliary instance, called the *baseline*, to represent the absence of features. This is different from previous Shapley-value approachs for trees [16,20], where the absence of features is modeled by the expectation of model predictions under some fixed features [22].
- We further develop a polynomial approach *BGShapTree* to compute BGShapvalues for tree models. The basic idea is to decompose the BGShapvalue into weights of

Received January 29, 2024; accepted May 7, 2024

E-mail: gaow@nju.edu.cn

leaves, and calculate weights by exploiting relations between features and leaves. Based on this approach, we could greedily search *salient feature groups* of large BGShapvalues. Our strategy is to firstly search salient feature groups with relatively small sizes, and then search larger salient feature groups iteratively.

- We finally conduct extensive experiments to validate the effectiveness of our approach. To be specific, our approach takes significantly better performance on twenty benchmark datasets, and presents more intuitive explanations on image data, compared with state-of-the-art methods on interpreting tree models.

The rest of this work is organized as follows. Section 2 presents BGShapvalue for the importance of a feature group with some desirable properties of interpretability. Section 3 presents our polynomial BGShapTree approach to calculate BGShapvalues for tree-based models, followed with a greedy algorithm to search salient feature groups. Section 4 presents related works. Section 5 presents some extensive experiments. Section 6 concludes this work.

2 Baseline group shapley value

We begin with some notations in this work. Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the instance space with feature set $[d] = \{1, 2, \ldots, d\}$. Denote by $\mathcal{F}: \mathcal{X} \to \mathbb{R}$ a model space, and assume that $f \in \mathcal{F}$ is a target model to be interpreted. Let $\mathbb{I}[\cdot]$ be the indicator function, which returns 1 if the input is true, and 0 otherwise. For set S, we use $|S|$ to denote its cardinality. We write $[k] = \{1, 2, \ldots, k\}$ for integer $k > 0$, and let $\emptyset$ stand for the empty set. For real number $r > 0$, let $\lceil r \rceil$ be the smallest integer not less than r.

The goal of this work is to explain the prediction of f on target instance $\boldsymbol{x}^* \in \mathcal{X}$. It is necessary to introduce an auxiliary instance $\boldsymbol{x}' \in \mathcal{X}$, called the *baseline*, to model the absence of features. For simplicity, we suppress the dependency on the choice of $\boldsymbol{x}^*$ and $\boldsymbol{x}'$ when we write related functions.

Motivated from game theory [23,24], we introduce the *Baseline Group Shapley Value* (*BGShapvalue*) to quantify the importance of a feature group $T \subseteq [d]$ as follows:

$$\phi_f(T) = \sum_{S \subseteq [d] \setminus T} p_d^{|T|}(|S|)\big(f(c(S \cup T)) - f(c(S))\big), \tag{1}$$

where $p_d^{|T|}(k)$ is given by

$$p_d^{|T|}(k) = \frac{k!(d - |T| - k)!}{(d - |T| + 1)!}, \tag{2}$$

and $c(S) = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_d)$ is defined as

$$\hat{x}_k = \begin{cases} x_k^*, & \text{for } k \in S, \\ x_k', & \text{for } k \in [d] \setminus S. \end{cases} \tag{3}$$

Intuitively speaking, $c(S)$ is a combination of target instance $\boldsymbol{x}^*$ and baseline $\boldsymbol{x}'$, which models features' absence by switching their corresponding values to those of baseline $\boldsymbol{x}'$.

From the view of game theory, we could regard feature group T and other features in $[d] \setminus T$ as $d - |T| + 1$ players, and these players are ordered randomly. BGShapvalue then essentially counts the average contribution of feature group T to those players that precede T in the orderings.

The BGShapvalue has some desirable properties of interpretability as mentioned in [21,25]. For independent models $f \in \mathcal{F}$ and $g \in \mathcal{F}$, we have

$$\phi_{f+g}(T) = \phi_f(T) + \phi_g(T) \text{ for any } T \subseteq [d].$$

This is known as *linearity axiom*, which helps us compute BGShapvalues for ensembles efficiently.

Given feature group $T \subseteq [d]$, if $f(c(S \cup T)) = f(c(S))$ for each $S \subseteq [d] \setminus T$, then we have

$$\phi_f(T) = 0.$$

This is known as *dummy axiom*, which could help us identify feature groups with no contribution.

Given $f \in \mathcal{F}$ and $g \in \mathcal{F}$, if $g(\boldsymbol{x}) = f(\boldsymbol{x})$ for any $\boldsymbol{x} \in \mathcal{X}$, then we have

$$\phi_f(T) = \phi_g(T) \text{ for any } T \subseteq [d].$$

This is known as *implementation invariance axiom*, that is, we provide same interpretations for two identical models with different implementations.

For two different features $i, j \in [d]$, if $f(c(S \cup \{i\})) = f(c(S \cup \{j\}))$ for every $S \subseteq [d] \setminus \{i, j\}$, then we have

$$\phi_f(\{i\}) = \phi_f(\{j\}),$$

which is known as *symmetry axiom*. This axiom ensures that we assign identical importance values for two symmetric individual features.

For individual features $i \in [d]$, we also have

$$\sum_{i \in [d]} \phi_f(\{i\}) = f(\boldsymbol{x}^*) - f(\boldsymbol{x}').$$

This is known as *efficiency axiom*, that is, the sum of BGShapvalues for all individual features equals the difference between $f(\boldsymbol{x}^*)$ and $f(\boldsymbol{x}')$.

A feature group $U \subseteq [d]$ is said to be a *carrier* if $f(c(S)) = f(c(S \cap U))$ for any $S \subseteq [d]$. We have *carrier axiom* as follows, which helps us identify salient feature groups without redundant features.

Lemma 1 For any carrier U and feature group $T \subseteq [d]$, we have $\phi_f(T) = \phi_f(T \cap U)$.

Proof From carrier U and Eq. (1), we have

$$\phi_f(T) = \sum_{S \subseteq [d] \setminus T} p_d^{|T|}(|S|) \big(f(c((S \cup T) \cap U)) - f(c(S \cap U))\big), \tag{4}$$

and we similarly have

$$\phi_f(T \cap U) = \sum_{S \subseteq [d] \setminus (T \cap U)} p_d^{|T \cap U|}(|S|) \big(f(c((S \cup T) \cap U)) - f(c(S \cap U))\big). \tag{5}$$

Notice that $S \cap U = (S \cup S') \cap U$ for every $S' \subseteq T \setminus U$, and this follows that

$$\phi_f(T \cap U) = \sum_{S \subseteq [d] \setminus T} \sum_{S' \subseteq (T \setminus U)} p_d^{|T \cap U|}(|S \cup S'|) \big(f(c((S \cup T) \cap U)) - f(c(S \cap U))\big). \tag{6}$$

We set $k=|S'|\in\{0,1,\ldots,|T\setminus U|\}$, and for $S\subseteq[d]\setminus T$, we have

$$\phi_f(T\cap U)=\sum_{S\subseteq[d]\setminus T}\sum_{k=0}^{|T\setminus U|}\binom{|T\setminus U|}{k}p_d^{|T\cap U|}(|S|+k)\\ \big(f(c((S\cup T)\cap U))-f(c(S\cap U))\big)\,. \tag{7}$$

Based on Proposition 1, we have

$$\sum_{k=0}^{|T\setminus U|}\binom{|S|+k}{|S|}\binom{d-|T\cap U|-|S|-k}{d-|T|-|S|}=\binom{d-|T\cap U|+1}{d-|T|+1}.$$

This follows that, from $|T|=|T\setminus U|+|T\cap U|$

$$\sum_{k=0}^{|T\setminus U|}\binom{|T\setminus U|}{k}\binom{d-|T|}{|S|}=\frac{(d-|T\cap U|+1)\binom{d-|T\cap U|}{|S|+k}}{d-|T|+1},$$

and we further have, from Eq. (2)

$$\sum_{k=0}^{|T\setminus U|}\binom{|T\setminus U|}{k}p_d^{|T\cap U|}(|S|+k)=p_d^{|T|}(|S|)\,,$$

which proves $\phi_f(T)=\phi_f(T\cap U)$ from Eqs. (4) and (7). □

Proposition 1 For non-negative integers p,q and n, we have

$$\sum_{k=0}^{n}\binom{p+k}{p}\binom{q+n-k}{q}=\binom{p+q+n+1}{p+q+1}.$$

Proof From generalized binomial theorem, we have

$$\frac{1}{(1-x)^{p+q+2}}=\sum_{n=0}^{\infty}\binom{p+q+n+1}{p+q+1}x^n\,. \tag{8}$$

On the other hand, we have

$$\frac{1}{(1-x)^{p+1}(1-x)^{q+1}}=\sum_{k=0}^{\infty}\binom{p+k}{p}x^k\sum_{k=0}^{\infty}\binom{q+k}{q}x^k,$$

and this follows that

$$\frac{1}{(1-x)^{p+1}(1-x)^{q+1}}=\sum_{n=0}^{\infty}\sum_{k=0}^{n}\binom{p+k}{p}\binom{q+n-k}{q}x^n,$$

which completes the proof by Eq. (8) □

The baseline Shapley value has been well-studied over individual features in previous works [21,26]. Another related work is the conditional expectation Shapley value [22,27,28], which considered the expectation of model predictions under some fixed features. Sundararajan and Najmi [21] give more details on the comparison between these two kinds of Shapley-value methods.

3 Our BGShapTree approach

We consider a single axis-parallel tree predictor f as our target model, and our approach could be directly extended to tree ensembles such as random forests according to linearity axiom. From Eq. (1), we could observe the $O(2^d h)$ time on the calculation of BGShapvalue for tree f with depth h and feature dimension d, and it requires high computation especially for large d. For this problem, our solution is to firstly decompose the BGShapvalue into weights over leaves, and then explore the correlations between features and leaves.

3.1 Decomposition of BGShapvalue

We assume that tree predictor f has m leaves, which are associated with m rectangular cells $C_1,C_2,\ldots,C_m$. Each rectangular cell C_j can be further expressed by d intervals $I_{1,j},\ldots,I_{d,j}$ as follows:

$$C_j=I_{1,j}\times\cdots\times I_{d,j}=\{\boldsymbol{x}=(x_1,\ldots,x_d)\colon x_k\in I_{k,j}\}\,.$$

Let L_j denote the set of features for splitting on the decision path of C_j, and features from $[d]\setminus L_j$ are irrelevant to the construction of C_j. Recall that target instance $\boldsymbol{x}^*=(x_1^*,\ldots,x_d^*)$ and baseline $\boldsymbol{x}'=(x_1',\ldots,x_d')$, and we have

$$x_k^*\in I_{k,j}\quad\text{and}\quad x_k'\in I_{k,j}\quad\text{for } k\in[d]\setminus L_j\,. \tag{9}$$

Denote by $T_j=T\cap L_j$, and it follows that

$$x_k^*\in I_{k,j}\quad\text{and}\quad x_k'\in I_{k,j}\quad\text{for } k\in T\setminus T_j\,. \tag{10}$$

We use r_j to denote the output of model f over the rectangular cell C_j for $j\in[m]$, that is, $r_j=f(\boldsymbol{x})$ for any $\boldsymbol{x}\in C_j$. For any $\boldsymbol{x}\in\mathcal{X}$, we have

$$f(\boldsymbol{x})=\sum_{j\in[m]}r_j\times\mathbb{I}[\boldsymbol{x}\in C_j]\,. \tag{11}$$

Based on Eq. (11), we could rewrite Eq. (1) as:

$$\phi_f(T)=\sum_{j\in[m]}w_{j,T}\times r_j\,,$$

where $w_{j,T}$ corresponds to the weight of the jth leaf, and it could be given by

$$w_{j,T}=\sum_{S\subseteq[d]\setminus T}p_d^{|T|}(|S|)(\mathbb{I}[c(S\cup T)\in C_j]-\mathbb{I}[c(S)\in C_j])\,.$$

This remains to calculate weights $\{w_{j,T}\}_{j\in[m]}$, and we present an example to illustrate the decomposition of BGShapvalue.

Example Let feature dimension $d=3$, and leaf number $m=4$. We consider target instance $\boldsymbol{x}^*=(0.4,0.2,0.6)$ and baseline $\boldsymbol{x}'=(0,0,0)$. For target group $T=\{1,2\}$, we have

$$\phi_f(T)=w_{1,T}r_1+w_{2,T}r_2+w_{3,T}r_3+w_{4,T}r_4\,,$$

where $w_{j,T}$ for $j\in[4]$ could be calculated by

$$w_{j,T}=p_3^2(0)(\mathbb{I}[(0.4,0.2,0)\in C_j]-\mathbb{I}[(0,0,0)\in C_j])\\ +p_3^2(1)(\mathbb{I}[(0.4,0.2,0.6)\in C_j]-\mathbb{I}[(0,0,0.6)\in C_j])\,.$$

3.2 Calculation on weights of leaves

For the jth leaf and corresponding rectangular cell C_j ($j\in[m]$), we first introduce two feature sets as:

$$A_j=\{k\in L_j\colon x_k^*\in I_{k,j}\}\,, \tag{12}$$

$$B_j=\{k\in L_j\colon x_k'\in I_{k,j}\}\,. \tag{13}$$

Here, A_j and B_j denote the subsets of features in L_j, whose values of $\boldsymbol{x}^*$ and $\boldsymbol{x}'$ fall in the corresponding intervals of C_j.

If $A_j\cup B_j\neq L_j$, then there is a $k\in L_j\setminus(A_j\cup B_j)$ such that $x_k^*\notin I_{k,j}$ and $x_k'\notin I_{k,j}$. This follows that $c(S)\notin C_j$ for any

$S \subseteq [d]$, and we have

$$w_{j,T} = 0 . \tag{14}$$

For $A_j \cup B_j = L_j$, we introduce a necessary and sufficient condition for $c(S) \in C_j$, which plays an important role in the subsequent analysis.

Theorem 1 For $A_j \cup B_j = L_j$, we have $c(S) \in C_j$ if and only if

$$S = (A_j \setminus B_j) \cup S' \quad \text{with} \quad S' \subseteq D_j , \tag{15}$$

where $D_j = (A_j \cap B_j) \cup ([d] \setminus L_j)$.

Proof We first prove $c(S) = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_d) \in C_j$ when Eq. (15) holds. Combining Eqs. (3) and (15), we have

$$\hat{x}_k = \begin{cases} x_k^*, & \text{for } k \in (A_j \setminus B_j) \subseteq S , \\ x'_k, & \text{for } k \in (B_j \setminus A_j) \subseteq ([d] \setminus S) . \end{cases}$$

This follows that from Eqs. (12) and (13)

$$\hat{x}_k \in I_{k,j} \quad \text{for} \quad k \in (A_j \setminus B_j) \cup (B_j \setminus A_j) .$$

We also have $\hat{x}_k = x_k^*$ or $\hat{x}_k = x'_k$ for $k \in D_j$, and this yields that

$$\hat{x}_k \in I_{k,j} \quad \text{for} \quad k \in D_j$$

from Eqs. (9), (12) and (13). We finally have

$$\hat{x}_k \in I_{k,j} \quad \text{for} \quad k \in [d],$$

from $A_j \cup B_j = L_j$, which proves that $c(S) \in C_j$.

If $c(S) \in C_j$, then we have $\hat{x}_k = x_k^*$ for $k \in A_j \setminus B_j$ from $x'_k \notin I_{k,j}$. This follows that $A_j \setminus B_j \subseteq S$ from Eq. (3).

We similarly have $\hat{x}_k = x'_k$ for $k \in B_j \setminus A_j$, and this could yield that

$$B_j \setminus A_j \subseteq [d] \setminus S,$$

from Eq. (3). Hence, we have

$$S = (A_j \setminus B_j) \cup S' \text{ with } S' \subseteq [d] \setminus ((A_j \setminus B_j) \cup (B_j \setminus A_j)) .$$

Notice that $A_j \cup B_j = L_j$, and this follows that

$$S = (A_j \setminus B_j) \cup S' \text{ with } S' \subseteq D_j ,$$

which completes the proof. □

As shown in Fig. 1, it is sufficient to discuss four cases for A_j, B_j and T_j when $A_j \cup B_j = L_j$:

- If $T_j \subseteq A_j \cap B_j$, then we have $x_k^* \in I_{k,j}$ and $x'_k \in I_{k,j}$ for $k \in T$ from Eqs. (10), (12) and (13). Notice that $\mathbb{I}[c(S \cup T) \in C_j] = \mathbb{I}[c(S) \in C_j]$ for $S \subseteq [d] \setminus T$, and this follows that

$$w_{j,T} = 0 . \tag{16}$$

- If $T_j \nsubseteq A_j$ and $T_j \nsubseteq B_j$, then there exists $k \in T_j \subseteq T$ such that $x_k^* \notin I_{k,j}$ from Eqs. (12) and (13). It follows that

$$c(S \cup T) \notin C_j \quad \text{for} \quad S \subseteq [d] \setminus T .$$

We also have $(A_j \setminus B_j) \nsubseteq S$ for any $S \subseteq [d] \setminus T$ from $T_j \nsubseteq B_j$, and this yields that $c(S) \notin C_j$ from Theorem 1. Hence, we finally have

$$w_{j,T} = 0 . \tag{17}$$

- If $T_j \subseteq A_j$ and $T_j \nsubseteq B_j$, then we have $(A_j \setminus B_j) \nsubseteq S$ for every $S \subseteq [d] \setminus T$. This follows that $c(S) \notin C_j$ from Theorem 1, and we have

$$w_{j,T} = \sum_{S \subseteq [d] \setminus T} p_d^{|T|} (|S|) \mathbb{I}[c(S \cup T) \in C_j] .$$

It suffices to consider $S \subseteq [d] \setminus T$ with $c(S \cup T) \in C_j$. From Theorem 1, we have

$$S \cup T = (A_j \setminus B_j) \cup S' \text{ with } S' \subseteq D_j ,$$

and we also have, from $S \subseteq [d] \setminus T$,

$$S = (A_j \setminus (B_j \cup T_j)) \cup S' \text{ with } S' \subseteq (D_j \setminus T) .$$

Write $k = |S'|$ and $\tau = |D_j \setminus T|$. We have $\binom{\tau}{k}$ different selections on S', and obtain that

$$w_{j,T} = \sum_{k=0}^{\tau} \binom{\tau}{k} p_d^{|T|} (|A_j \setminus (B_j \cup T_j)| + k) . \tag{18}$$

- If $T_j \subseteq B_j$ and $T_j \nsubseteq A_j$, then there exists $k \in T_j \subseteq (S \cup T)$ such that $k \in B_j \setminus A_j$. This follows that $c(S \cup T) \notin C_j$, and we have

$$w_{j,T} = - \sum_{S \subseteq [d] \setminus T} p_d^{|T|} (|S|) \mathbb{I}[c(S) \in C_j] .$$

Similarly to Eq. (18), we have, from $S = (A_j \setminus B_j) \cup S'$ with $S' \subseteq (D_j \setminus T)$,

$$w_{j,T} = - \sum_{k=0}^{\tau} \binom{\tau}{k} p_d^{|T|} (|A_j \setminus (B_j \cup T_j)| + k) . \tag{19}$$

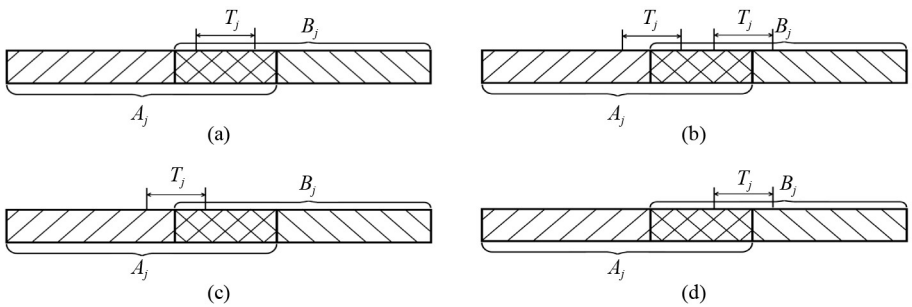

Fig. 1 An illustration of four cases for A_j, B_j and T_j when $A_j \cup B_j = L_j$. (a) $T_j \subseteq A_j \cap B_j$; (b) $T_j \nsubseteq A_j$ and $T_j \nsubseteq B_j$; (c) $T_j \subseteq A_j$ and $T_j \nsubseteq B_j$; (d) $T_j \nsubseteq A_j$ and $T_j \subseteq B_j$

In Algorithm 1, we present a detailed description of our BGShapTree approach. It is observable that our approach takes at most $O(d)$ time to calculate the weight of one leaf since $\tau < d$ in Eqs. (18) and (19). We finally take at most $O(md)$ running time to calculate $\phi_f(T)$ for a given tree model f with m leaves. Our approach also takes at most $O(m+d)$ memory space to store $\{r_j\}_{j\in[m]}$ and some necessary feature sets for the calculation of $\phi_f(T)$.

In practice, we could speed up our BGShapTree approach by storing some weights $\{w_{j,T}\}$ in advance if we aim to interpret multiple target instances. This is because there are at most $(d+1)^2$ selections for $\{w_{j,T}\}$ owing to $0 \leqslant \tau \leqslant d$ and $0 \leqslant |A_j \setminus (B_j \cup T_j)| \leqslant d$ for any $f \in \mathcal{F}$ and $\boldsymbol{x}^* \in \mathcal{X}$. For simplicity, we introduce following notation:

$$q_d^{|T|}(u,v) = \sum_{k=0}^{u} \binom{u}{k} p_d^{|T|}(v+k) \,. \tag{20}$$

Then Eqs. (18) and (19) could be rewritten as

$$w_{j,T} = q_d^{|T|}(\tau, |A_j \setminus (B_j \cup T_j)|) \,, \tag{21}$$

$$w_{j,T} = -q_d^{|T|}(\tau, |A_j \setminus (B_j \cup T_j)|) \,. \tag{22}$$

From Eqs. (21) and (22), it is sufficient to obtain $q_d^{|T|}(u,v)$ with $0 \leqslant u, v \leqslant d$ for the calculation of $w_{j,T}$. We present following recursive relation when computing $q_d^{|T|}(u,v)$.

Lemma 2 For integers $1 \leqslant u \leqslant d$ and $0 \leqslant v \leqslant d-1$, we have

$$q_d^{|T|}(u,v) = q_d^{|T|}(u-1,v) + q_d^{|T|}(u-1,v+1) \,.$$

Proof From Eq. (20), we have

$$\begin{aligned} q_d^{|T|}(u,v) &= \sum_{k=0}^{u} \binom{u-1}{k} p_d^{|T|}(v+k) \\ &\quad + \sum_{k=0}^{u} \binom{u-1}{k-1} p_d^{|T|}(v+k) \,. \end{aligned} \tag{23}$$

On the other hand, we have

$$\begin{aligned} q_d^{|T|}(u-1,v) &= \sum_{k=0}^{u-1} \binom{u-1}{k} p_d^{|T|}(v+k) \\ &= \sum_{k=0}^{u} \binom{u-1}{k} p_d^{|T|}(v+k) \,, \end{aligned} \tag{24}$$

and we also have

$$\begin{aligned} q_d^{|T|}(u-1,v+1) &= \sum_{k=0}^{u-1} \binom{u-1}{k} p_d^{|T|}(v+1+k) \\ &= \sum_{k=0}^{u} \binom{u-1}{k-1} p_d^{|T|}(v+k) \,. \end{aligned}$$

This completes the proof from Eqs. (23) and (24). □

From Lemma 2, we could take at most $O(d^2)$ time to compute and store $q_d^{|T|}(u,v)$ for $0 \leqslant u \leqslant d$ and $0 \leqslant v \leqslant d$ by recursive calculation. After storage, it remains the $O(mh)$ running time to calculate the BGShapvalue $\phi_f(T)$ for tree f of depth h since it suffices to scan the decision paths of m leaves.

3.3 Search the salient feature group

In Algorithm 1, we present an efficient calculation on the BGShapvalue of a given feature group $T \subseteq [d]$. In many real applications, we try to find a feature group of the largest BGShapvalue, which is called the *salient feature group*. Obviously, it is rather intractable to exactly search the salient feature group from the total 2^d feature subsets.

We present a greedy method to search the salient feature group based on Algorithm 1. Generally speaking, the salient feature group can be composed of important groups with smaller sizes, and we can firstly search salient individual features and search larger feature groups iteratively. We terminate this process after $t-1$ iterations, where t is a pre-defined hyper-parameter denoting the expected size of the output salient feature group.

We introduce a list P of size κ to store potential salient feature groups, and apply Algorithm 1 on each individual feature to initialize P as

$$P = \{\{k_1\}, \ldots, \{k_\kappa\}\} \,, \tag{25}$$

where $k_1, k_2, \ldots, k_\kappa$ denote indices of features with the largest κ BGShapvalues.

For each iteration, we first construct a temporary set P' of

Algorithm 1 The BGShapTree algorithm

Input: Target instance $\boldsymbol{x}^*$, baseline $\boldsymbol{x}'$, target set T and tree predictor f

Output: BGShapvalue $\phi_f(T)$

Initialize: $w_{j,T} = 0$ for $j \in [m]$ and $\phi_f(T) = 0$

for $j = 1, 2, \ldots, m$ **do**
  Compute sets A_j and B_j by Eqs. (12)–(13)
  if $A_j \cup B_j = L_j$ and $T_j \subseteq A_j$ and $T_j \nsubseteq B_j$ **then**
    Compute weight $w_{j,T}$ by Eq. (18)
  end if
  if $A_j \cup B_j = L_j$ and $T_j \subseteq B_j$ and $T_j \nsubseteq A_j$ **then**
    Compute weight $w_{j,T}$ by Eq. (19)
  end if
  Compute $\phi_f(T) = \phi_f(T) + r_j \times w_{j,T}$
end for
return $\phi_f(T)$

Algorithm 2 Search for salient feature group

Input: Target instance $\boldsymbol{x}^*$, baseline $\boldsymbol{x}'$, tree predictor f, parameter t and κ

Output: Salient feature group T^*

Initialize P according to Eq. (25) by applying Algorithm 1 on every individual feature
for $j = 1, 2, \ldots, t-1$ **do**
  Calculate P' according to Eq. (26)
  Calculate the BGShapvalue $\phi_f(T)$ for $T \in P'$ based on Algorithm 1
  Update P with feature groups of the largest κ BGShapvalues in P'
end for
return $T^* = \arg\max_{T \in P} \phi_f(T)$

extended feature groups as follows:

$$P' = \{T \cup \{i\} \colon T \in P, i \in [d] \setminus T\} . \tag{26}$$

We then calculate the BGShapvalue for each $T \in P'$ based on Algorithm 1, and update P with feature groups of the largest κ BGShapvalues. We finally output the salient feature group in P of the largest BGShapvalue after $t-1$ iterations. Algorithm 2 presents the detailed descriptions for such process.

In Algorithm 2, we could speed up the calculation on BGShapvalues of feature groups by storing some important variables. For simplicity, we introduce some notations as

$$u_{j,T} = |D_j \setminus T| \quad \text{and} \quad v_{j,T} = |A_j \setminus (B_j \cup T_j)| ,$$

which play important roles on the calculation of $w_{j,T}$ according to Eqs. (21) and (22).

We could store $u_{j,T}$ and $v_{j,T}$ when computing $w_{j,T}$. For extended feature group $T' = T \cup \{i\}$, we have $u_{j,T'} = u_{j,T} - 1$ for $i \in (A_j \cap B_j) \cup ([d] \setminus L_j)$, and $u_{j,T'} = u_{j,T}$ otherwise. We also have $v_{j,T'} = v_{j,T} - 1$ for $i \in A_j \setminus B_j$, and $v_{j,T'} = v_{j,T}$ otherwise.

From Eq. (14), it is sufficient to consider the case of $A_j \cup B_j = L_j$, and we could calculate $w_{j,T'}$ based on the relation between T_j and i as:

- We have $T'_j \subseteq A_j$ and $T'_j \nsubseteq B_j$ if $T_j \subseteq A_j \cap B_j$ and $i \in A_j \setminus B_j$, or $T_j \subseteq A_j$ and $T_j \nsubseteq B_j$ and $i \in A_j \cup ([d] \setminus L_j)$. It follows that, from Eqs. (18) and (21),

$$w_{j,T'} = q_d^{|T'|}(u_{j,T'}, v_{j,T'}) .$$

- We have $T'_j \subseteq B_j$ and $T'_j \nsubseteq A_j$ if $T_j \subseteq A_j \cap B_j$ and $i \in B_j \setminus A_j$, or $T_j \nsubseteq A_j$ and $T_j \subseteq B_j$ and $i \in B_j \cup ([d] \setminus L_j)$. From Eqs. (19) and (22), we could obtain that

$$w_{j,T'} = -q_d^{|T'|}(u_{j,T'}, v_{j,T'}) .$$

- For other cases, we have $T'_j \subseteq A_j \cap B_j$ or $T'_j \nsubseteq A_j \wedge T'_j \nsubseteq B_j$. It follows that $w_{j,T'} = 0$ from Eqs. (16) and (17).

Hence, we could take $O(1)$ time to calculate $w_{j,T'}$ after storing $u_{j,T}$ and $v_{j,T}$. We finally take at most $O(m)$ running time to compute $\phi_f(T')$, which could significantly speed up the process of Algorithm 2.

Algorithm 2 requires calculating BGShapvalues of all d individual features, and we could observe the $O(md^2)$ time by applying Algorithm 1 directly. In practice, we could improve this to $O(md)$ level. Similarly to Algorithm 1, we have

$$w_{j,\{i\}} = \sum_{S \subseteq [d] \setminus \{i\}} p_d^{\{i\}}(|S|)(\mathbb{I}[c(S \cup \{i\}) \in C_j] - \mathbb{I}[c(S) \in C_j])$$

for $i \in [d]$ and $j \in [m]$. For $i \notin L_j$, we have

$$\mathbb{I}[c(S \cup \{i\}) \in C_j] = \mathbb{I}[c(S) \in C_j] ,$$

and it follows that $w_{j,\{i\}} = 0$. It remains to calculate $w_{j,\{i\}}$ for $i \in L_j$. From Eq. (14), it is sufficient to study $A_j \cup B_j = L_j$. We first have $w_{j,\{i\}} = 0$ for $i \in A_j \cap B_j$ from Eq. (16). For $i \in A_j \setminus B_j$, we have

$$w_{j,\{i\}} = \sum_{k=0}^{\tau} \binom{\tau}{k} p_d^{\{i\}}(|A_j \setminus B_j| + k - 1)$$

from Eq. (18), where $\tau = |A_j \cap B_j| + d - |L_j|$. For $i \in B_j \setminus A_j$, we similarly have

$$w_{j,\{i\}} = -\sum_{k=0}^{\tau} \binom{\tau}{k} p_d^{\{i\}}(|A_j \setminus B_j| + k)$$

from Eq. (19), where $\tau = |A_j \cap B_j| + d - |L_j|$.

From the above discussion, we notice that there are at most two different non-zero selections for $\{w_{j,\{i\}}\}_{i \in [d]}$, and they can be computed in $O(d)$ time. This yields an $O(md)$ running time to calculate the BGShapvalues for all d individual features. This complexity can be further improved to $O(mh)$ level after we calculate and store different weights in memory as described in Section 3.2.

4 Related work

On the global interpretation of tree models over a data set, Breiman [9] considered the reduction of impurity during splitting as the measure of feature importance. Diaz and Alvarez [29] permuted the value of a feature and computed its importance by observing the change on the model's error. Strobl et al. [10] derived feature importance based on the splitting number of a feature. Along this line, much attention has been paid on the interpretations of tree-based models from both empirical [30–34] and theoretical views [14,35,36].

Sagi and Rokach [37] transformed complex tree ensembles into interpretable trees. Tan et al. [38] used representative instances to interpret trees. Counterfactual interpretations have also been presented for tree models [39–41]. Some studies presented logical methods to interpret trees [42–45] and Agarwal et al. [46] presented a regularization to improve the interpretability of tree models.

For the local interpretation of tree models over one target instance (i.e., a single prediction), Saabas [12] computed feature importance according to the decision path of the target instance. Several works used the Shapley value to interpret an individual prediction for tree models [16,20,47]. Sutera et al. [19] generalized impurity reduction to interpret a single prediction. Besides, XGBoost has also been interpreted over a single instance [17,18].

The Shapley value has played an important role towards the interpretation of other models such as linear models [48–50], kernel methods [51], deep neural networks [26,52–56], black-box models [57–64] and so on. In addition, Wang et al. [65] and Beechey et al. [66] also applied Shapley values to interpret reinforcement learning, and Ren et al. [67] applied Shapley values to interpret adversarial learning. Chau et al. [68] and Watson et al. [69] adopted Shapley values to explain predictive uncertainty.

Theoretical studies have been presented for interpretation methods based on Shapley values. Frye et al. [60] established a theoretical framework for Shapley values to incorporate causal knowledge into interpretability. Sundararajan et al. [62] established the relationships between Shapley values and Taylor series. Janzing et al. [70] and Sundararajan and Najmi [21] compared different settings of Shapley-value methods

theoretically. Several studies analyzed the drawbacks of Shapley values as feature importance [71–73]. Van den Broeck et al. [74] analyzed the computational complexity of some Shapley-value methods. Bordt and von Luxburg [75] also studied the relationship between Shapley values and generalized additive models.

Towards the interpretation over multiple features, Jullum et al. [76] extended the conditional expectation Shapley value to measure the importance of some pre-defined feature groups, and such approach has been applied in various applications [20,77,78]. Our work extends the baseline Shapley value to feature groups, which presents intuitive interpretations and is flexible to model the explanation context with respect to different applications as mentioned in previous study [21].

5 Experiments

In this section, we conduct extensive experiments to validate the effectiveness of our approaches. We begin with some experimental settings, and followed with an illustrative example. We empirically compare our approach with state-of-the-art methods on the interpretations of tree ensembles, and show some interpretation visualization on image datasets. We then analyze different settings of our approach. We finally conduct additional experiments towards the interpretation of a single decision tree.

5.1 Experimental settings

In this work, we conduct experiments on twenty benchmark datasets provided by OpenML [79] and UCI [80], and relevant details are summarized in Table 1. Most of these datasets have been applied in previous studies, and all the features have been scaled to $[0,1]$ for all datasets. For all interpreted tree models, we adopt gini index as the splitting criterion.

We compare our approach with six state-of-the-art tree-specific methods on the interpretation of individual features, and one black-box method on the interpretation of feature groups as follows:

- TreeSHAP: An interpretation method based on the estimation of conditional expectation Shapley values [16];
- ACVTree: Another interpretation method on the estimation of conditional expectation Shapley values [20];
- TreeInterpret: An interpretation approach on the changes of average outputs over the decision path for the target instance [12];
- LocalMDI: An interpretation approach based on the decrease of impurity on the decision path for the target instance [19];
- GlobalMDI: An interpretation approach on the decrease of impurity over all nodes [9];
- SplitCount: An interpretation approach on the splitting number of a feature [10];
- GroupSHAP: An interpretation method based on the extension of conditional expectation Shapley values for feature groups [76];

We focus on three baselines for our approach: average instance over the training data, instance of all-zero features and instance of all-one features. We take the average BGShapvalue over these baselines as our importance measure.

Table 1 Benchmark datasets

Datasets	#inst	#feat	Datasets	#inst	#feat
Diabetes	520	16	Har	10,299	562
Australia	690	14	Pendigits	10,992	16
Vehicle	946	18	Drybean	13,661	16
Collins	1,000	23	Eggeye	14,980	14
Phishing	1,100	30	Magic04	19,020	10
Segment	2,310	19	Bank	45,200	16
Ginaprior	3,470	784	Shuttle	58,000	9
Texture	5,500	40	Sensor	58,509	48
Mushro	8,120	22	Mnist	60,000	784
Indian	9,144	220	Fmnist	60,000	784

We implement the LocalMDI method and the GroupSHAP method according to [19] and [76]. For other compared methods, we take the default parameter settings from their respective references. All experiments are performed with Python 3 on an Intel Core i9-10900X processor.

5.2 An illustrative example

We present an example to show different selections on salient feature groups between our method and other methods based on individual features, as well as the importance of our selections. For simplicity, we take TreeSHAP as a representative method of individual features, and similar results could also be observed for other methods. We train a random forest of 10 trees as our target model, and take a small dataset *diabetes* with binary features on the prediction of examples' likelihood.

We aim to interpret the target instance with 91% likelihood of positive label, as shown in Fig. 2. Our approach takes different selections on salient feature groups compared with TreeSHAP when the group cardinality ranges from 3 to 5; this is because our approach takes multiple features as an integral part, while TreeSHAP measures the importance of individual features independently.

We take the decrease of output likelihood (by setting zeros to salient features) as a metric for the importance of salient feature groups, similarly to [81]. As shown in Fig. 2, our selected feature groups are more important than those of TreeSHAP; for example, the output likelihood decreases by 66% after setting our selected features as zeros, whereas it decreases by 58% for TreeSHAP.

5.3 Experimental comparisons

For each dataset, we train 10 random trees as our interpreted models, and randomly choose 100 test instances as the target instances. We take $t = \lceil 0.3d \rceil$ for the size of target salient feature group. For GroupSHAP, we adopt its importance measure into Algorithm 2 to search salient feature groups. We also select parameter $\kappa = 10$ for the length of list P as shown in Algorithm 2. For other compared methods over individual features, we choose those features of the largest t importance values as their selected salient feature groups.

We take the test accuracies as our performance measure to evaluate selected salient feature groups, and maintain the salient features while replace all non-salient features with random features over uniform distribution on $[0,1]$; this is similar to the metrics in previous studies [16,82]. The final test

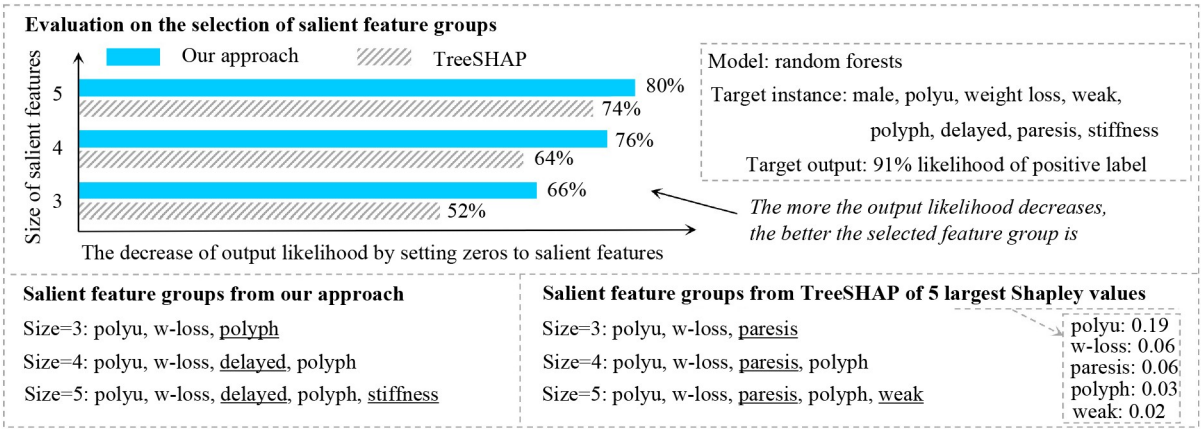

Fig. 2 Illustration of differences on salient feature groups between our approach and TreeSHAP (individual features)

accuracies are averaged over 100 random trials of non-salient features, as summarized in Table 2.

From Table 2, it is obviously observable that our approach achieves significantly better performance than other methods when their results are returned, since the win/tie/loss counts clearly show that our approach wins for most datasets and never losses. Moreover, our approach could also improve at least 3% of test accuracies on average in contrast to other compared methods. An intuitive explanation is that our approach tends to capture intrinsic structures and complex correlations among features.

The GroupSHAP approach does not return any result on most datasets after running out 72 hours because of its exponential running-time complexity on the calculation of Shapley values. We could also observe that TreeSHAP takes better performance than other methods over individual features due to the desirable properties of Shapley values.

We make comparisons on different cardinalities of salient feature groups, as shown in Fig. 3. Here, we present the experimental results on six representative datasets, and similar trends could be observed for other datasets. As shown in Fig. 3, our approach generally achieves better performance than other compared methods when $t \in [0.2d, 0.6d]$, which indicates that our approach could effectively capture correlations and structures among features in such range. It is also observable that our approach takes comparable performance with other methods as for $t \in [0.8d, d]$, since most important features have been selected for compared methods in such range.

We also present the running time of compared methods on ten benchmark datasets in Fig. 4, and similar results could also be observed on other datasets. As expected, global methods GlobalMDI and SplitCount are the fastest methods, since they are independent of specific instances. Among other local methods, LocalMDI and TreeInterpret take relatively low computational cost because they only need to scan the

Table 2 Comparisons of the testing accuracies (mean±std) for random forests classifiers. • indicates that our approach is significantly better than the corresponding methods for the selections of salient feature groups (pairwise t-test at 95% significance level). N/A means that the corresponding method does not return results within 72 hours

Datasets	Our approach	TreeSHAP	ACVTree	TreeInterpret	LocalMDI	GlobalMDI	SplitCount	GroupSHAP
Diabetes	.8382±.0331	.7924±.0318•	.8155±.0293•	.8171±.0309•	.7933±.0287•	.8021±.0309•	.7629±.0298•	N/A
Australia	.8709±.0113	.8591±.0189•	N/A	.8517±.0174•	.7965±.0298•	.8180±.0317•	.5077±.0505•	.8602±.0248•
Vehicle	.6480±.0324	.6276±.0303•	N/A	.5961±.0376•	.5361±.0291•	.4334±.0407•	.2956±.0395•	N/A
Collins	.9126±.0178	.8992±.0151•	.8848±.0208•	.8672±.0222•	.8258±.0291•	.7731±.0352•	.7822±.0302•	N/A
Phishing	.7573±.0219	.7449±.0249•	.7474±.0243	.7428±.0243•	.7165±.0267•	.7333±.0312•	.4647±.0486•	N/A
Segment	.7864±.0282	.7380±.0317•	N/A	.6651±.0357•	.6487±.0034•	.6406±.0348•	.3941±.0420•	N/A
Ginaprior	.9198±.0029	.9184±.0014	N/A	.8056±.0264•	.8266±.0215•	.7621±.0253•	.6397±.0347•	N/A
Texture	.6884±.0390	.6275±.0396•	5952±.0397•	.5815±.0410•	.5654±.0419•	.5114±.0445•	.4476±.0417•	N/A
Mushro	.9783±.0126	.9766±.0141	N/A	.9782±.0146	.9640±.0156•	.9195±.0223•	.8340±.0322•	N/A
Indian	.7806±.0348	.6646±.0365•	N/A	.7067±.0351•	.6823±.0390•	.5387±.0441•	.4908±.0435•	N/A
Har	.9796±.0020	.9718±.0075•	N/A	.9614±.0123•	.9580±.0097•	.6208±.0463•	.8058±.0330•	N/A
Pendigits	.6310±.0432	.5655±.0426•	.5233±.0411•	.4769±.0454•	.4689±.0435•	.2770±.0392•	.1755±.0347•	N/A
Drybean	.5112±.0444	.5030±.0377	.4487±.0407•	.4767±.0410•	.4665±.0439•	.4253±.0438•	.1488±.0.301•	N/A
Eggeye	.6579±.0337	.5547±.0393•	N/A	.5816±.0408•	.5123±.0348•	.5092±.0474•	.4971±.0232•	.6405±.0170•
Magic04	.5447±.0413	.5207±.0374•	.5172±.0396•	.4944±.0415•	.4737±.0339•	.4314±.0396•	.4540±.0362•	.5224±.0406•
Bank	.8356±.0271	.8163±.0271•	N/A	.8127±.0261•	.8304±.0273	.8089±.0282•	.8170±.0312•	N/A
Shuttle	.8524±.0267	.7830±.0381•	.8038±.0387•	.7738±.0340•	.7397±.0351•	.7347±.0341•	.5970±.0389•	.8322±.0221•
Sensor	.9184±.0239	.8886±.0256•	N/A	.8391±.0284•	.8154±.0340•	.7089±.0408•	.6597±.0367•	N/A
Mnist	.9499±.0010	.9504±.0002	N/A	.8957±.0178•	.8490±.0214•	.8743±.0146•	.8703±.0187•	N/A
Fmnist	.8518±.0015	.7529±.0146•	N/A	.7239±.0129•	.7143±.0012•	.2732±.0217•	.2261±.0229•	N/A
Average	.7955±.1372	.7578±.1503	–	.7324±.1518	.7092±.1551	.6298±.1891	.5435±.2203	–
Win/tie/loss		**16/4/0**	**20/0/0**	**19/1/0**	**19/1/0**	**20/0/0**	**20/0/0**	**20/0/0**

decision path of the target instance, while other methods require traversing all leaves. It is also observable that our approach takes less running time than ACVTree and GroupSHAP for all datasets as these two methods take exponential running-time to calculate Shapley values.

5.4 Interpretation visualization

We present the interpretation visualization on two image datasets *mnist* and *fmnist*. We also consider random forests of 10 random trees as our interpreted models. For simplicity, we set $t = 100$ for the cardinality of salient feature groups.

Figure 5 visualizes selected salient feature groups. It is evident that our approach makes better object recognition from the background, and present consistent interpretations with persons' intuitions. This is because our approach pays more attention to the structural information in images, rather than independent individual pixels.

ACVTree and GroupSHAP do not return results in 72 hours due to their exponential computational cost. LocalMDI and TreeInterpret make similar interpretations because they take similar measures on feature importance along the decision path of the input instance. GlobalMDI and SplitCount may not be suitable to interpret a single prediction since the identical result is returned for a given model, regardless of different target instances.

5.5 Analysis on baseline and parameter κ

We analyze different selections on baselines and parameter κ. We also take random forests of 10 random trees as our target models. For simplicity, we focus on three typical baseline instances: average instance over the training data, instance of all-zero features and instance of all ones and discuss seven combinations of them. For each combination, we take the average BGShapvalues as our importance measure, and then search salient feature groups based on Algorithm 2.

Figure 6 shows the empirical results of seven different combinations. It is observable that our approach with an average of three baselines outperforms other combinations in most datasets, and it is consistent with previous empirical results [64]. An intuitive explanation is that multiple baseline instances offer a more comprehensive and robust assessment of feature importance by capturing a wider range of feature impact and variability.

We also analyze the influence of parameter κ, which stands

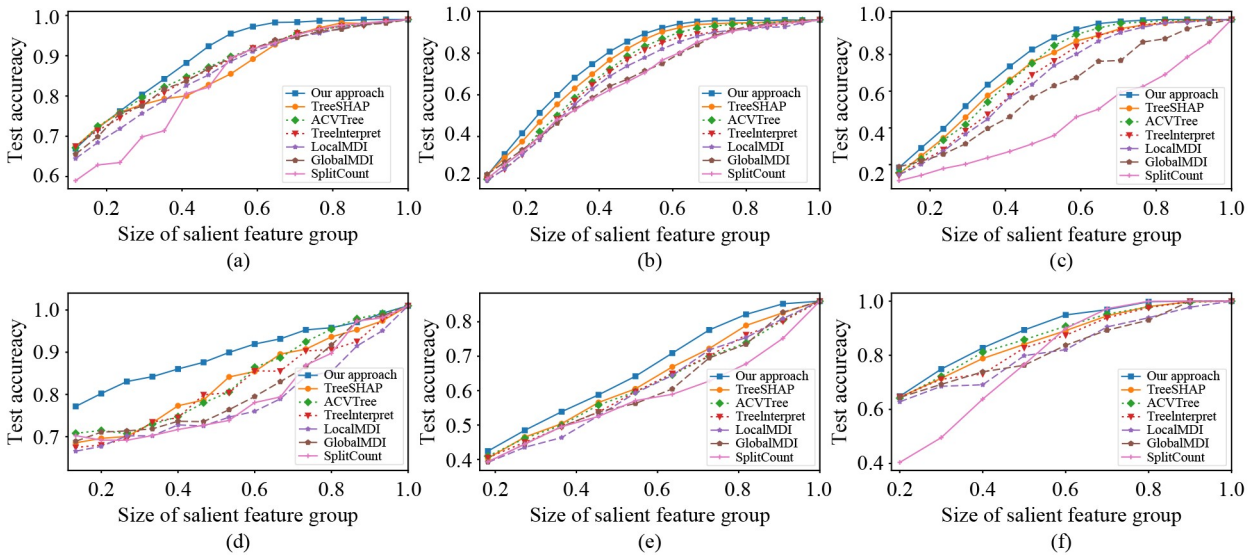

Fig. 3 Comparisons on the selections of salient feature groups with different cardinalities, where we scale the cardinality of salient feature groups to $[0, 1]$. The higher the curve, the better the performance. (a) Diabetes; (b) texture; (c) pendigits; (d) eggeye; (e) magic04; (f) shuttle

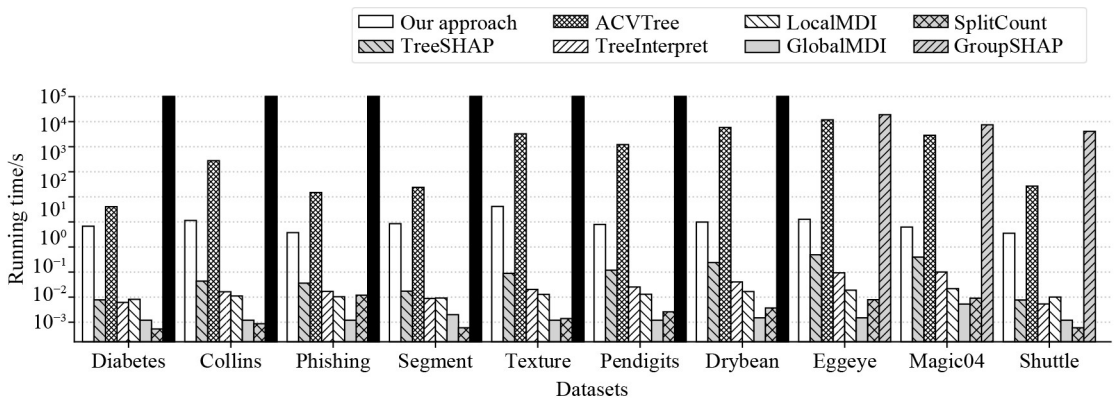

Fig. 4 Comparisons of the running time on 10 benchmark datasets (in seconds). Notice that the y-axis is in log-scale and full black columns imply that no result was obtained after running out 72 hours

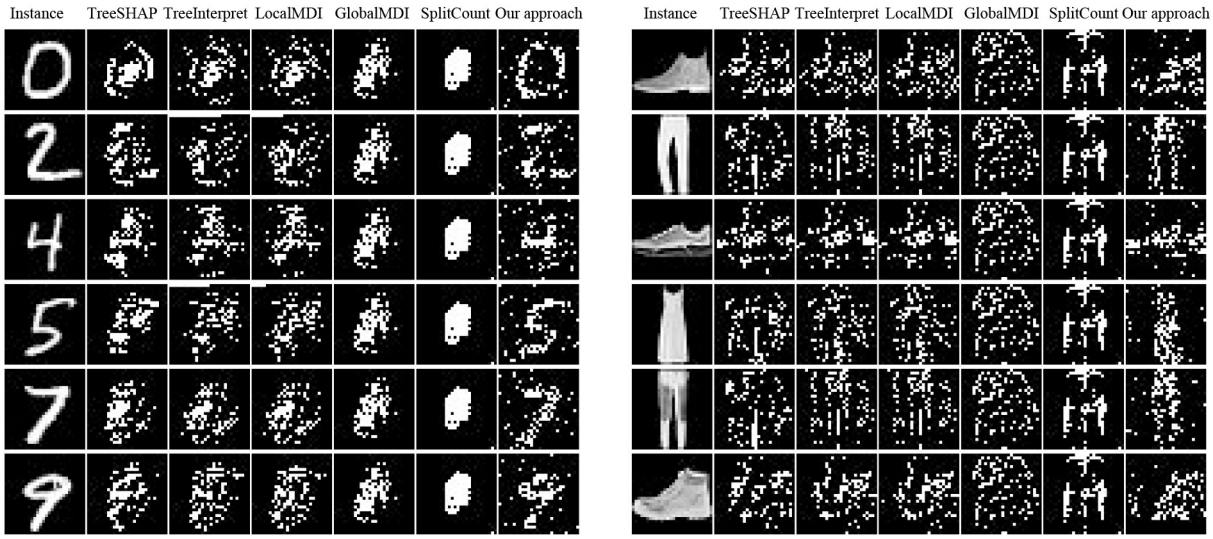

Fig. 5 Comparisons of interpretation visualization between our approach and other compared methods

for the size of temporary list P in Algorithm 2. For simplicity, we select parameter κ from $[1,2,5,10,20,50]$ and Fig. 7 presents the empirical results over these selections. As can be seen, our approach is not sensitive to the selection of parameter κ, and it generally works well for $\kappa \geqslant 5$ over all cardinalities of feature groups. This also confirms the validity of our choice $\kappa = 10$.

5.6 Interpretations of decision trees

We have already validated the effectiveness of our approach on the interpretations of tree ensembles, and this section makes additional experiments over a single decision tree. We train a single decision tree classifier as our target model, and randomly choose 100 test instances as the target instances. Similarly with Section 5.3, we take cardinality $t = \lceil 0.3d \rceil$ of target salient feature group, and the final test accuracies are averaged over 100 independent random trials, as summarized in Table 3.

From Table 3, we could easily observe that our approach also exhibits better performance on the interpretation of a single tree model compared with other methods, as the win/tie/loss counts clearly show that our approach wins for most datasets, yet only losses for sparse dataset *ginaprior* and class-imbalanced dataset *shuttle*. Moreover, our approach could improve at least 4% of test accuracies on average in contrast to other compared methods. This indicates that our approach could also capture correlations among features more effectively than other methods when applied to a single tree.

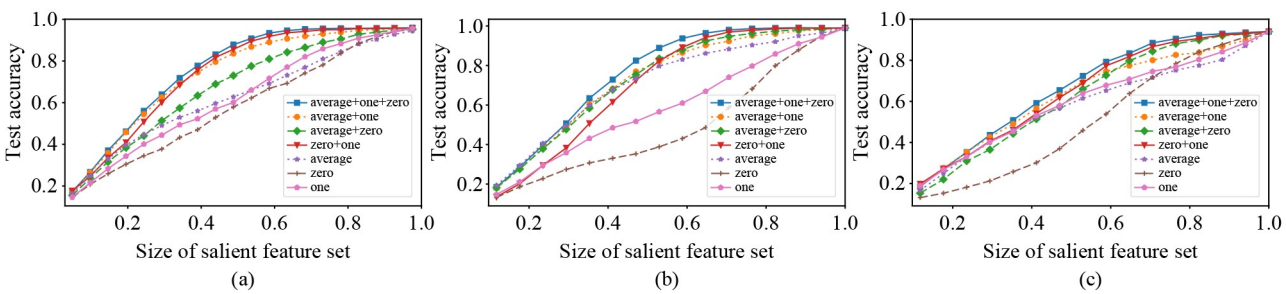

Fig. 6 Comparisons on different baseline combinations, where we scale the cardinality of salient feature groups to $[0,1]$. The higher the curve, the better the performance. (a) Texture; (b) pendigits; (c) drybean

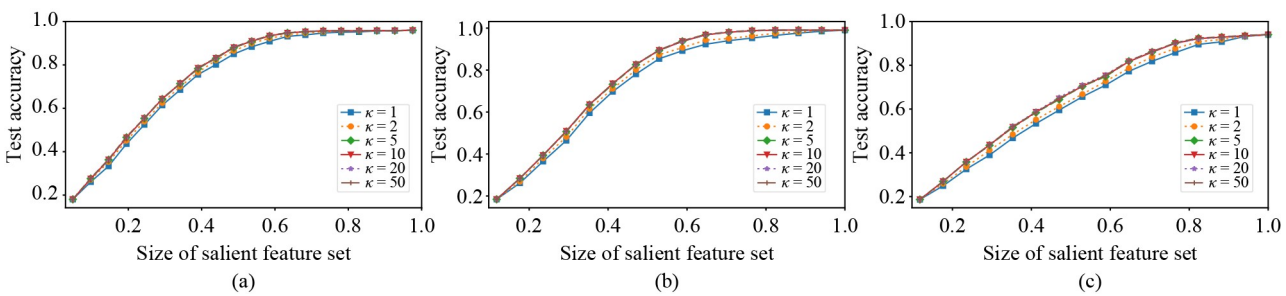

Fig. 7 Comparisons on different parameter κ, where we scale the cardinality of salient feature groups to $[0,1]$. The higher the curve, the better the performance. (a) Texture; (b) pendigits; (c) drybean

Table 3 Comparisons of the testing accuracies (mean±std) for individual decision trees. •/○ indicates that our approach is significantly better/worse than the corresponding methods for the selections of salient feature groups (pairwise t-test at 95% significance level). N/A means that the corresponding method does not return results within 72 hours

Datasets	Our approach	TreeSHAP	ACVTree	TreeInterpret	LocalMDI	GlobalMDI	SplitCount	GroupSHAP
Diabetes	.9316±.0145	.8334±.0296•	.8850±.0253•	.8593±.0310•	.8961±.0199•	.7803±.0312•	.6239±.0411•	.9173±.0017•
Australia	.8564±.0215	.8371±.0312•	.7119±.0348•	.8318±.0194•	.8588±.0160	.6798±.0338•	.5054±.0438•	.8293±.0155•
Vehicle	.6910±.0315	.6359±.0344•	.6001±.0328•	.6478±.0318•	.5651±.0341•	.4452±.0392•	.2739±.0388•	N/A
Collins	.9805±.0022	.9798±.0016	.9519±.0129•	.9755±.0046•	.9686±.0014•	.9655±.0182•	.9518±.0028•	N/A
Phishing	.7623±.0320	.7616±.0275	.7283±.0346•	.7343±.0299•	.7013±.0294•	.7124±.0337•	.4327±.0501•	N/A
Segment	.9353±.0102	.9043±.0152•	.8771±.0204•	.8774±.0151•	.9012±.0151•	.7707±.0216•	.3951±.0444•	N/A
Ginaprior	.8519±.0129	.8663±.0063○	N/A	.7750±.0270•	.7668±.0272•	.8704±.0045○	.8737±.0043○	N/A
Texture	.8592±.0138	.7990±.0273•	.8223±.0228•	.8265±.0202•	.8406±.0178•	.5336±.0375•	.5641±.0444•	N/A
Mushro	.9998±.0010	.9543±.0233•	.9568±.0223•	.9956±.0045•	.9926±.0065•	.9065±.0197•	.8981±.0279•	N/A
Indian	.7974±.0131	.7541±.0321•	N/A	.6723±.0376•	.6738±.0422•	.4709±.0415•	.5132±.0377•	N/A
Har	.9208±.0086	.9221±.0113	N/A	.8938±.0157•	.9077±.0149•	.9105±.0018•	.9026±.0025•	N/A
Pendigits	.4807±.0394	.4284±.0439•	.4481±.0383•	.3824±.0366•	.3891±.0331•	.3331±.0407•	.3077±.0413•	.4360±.0419
Drybean	.7089±.0338	.6561±.0471•	.5924±.0327•	.6647±.0331•	.5789±.0404•	.4923±.0404•	.1914±.0394•	.6871±.0266•
Eggeye	.6456±.0360	.6225±.0395•	.5439±.0352•	.6019±.0362•	.5467±.0379•	.5608±.0416•	.4536±.0428•	.6459±.0106
Magic04	.6639±.0461	.6232±.0437•	.5303±.0405•	.5398±.0441•	.5309±.0471•	.5051±.0445•	.5203±.0410•	.6220±.0429•
Bank	.6878±.0394	.5894±.0418•	.5656±.0439•	.4823±.0503•	.4931±.0466•	.5186±.0496•	.6141±.0403•	.6790±.0286•
Shuttle	.8721±.0305	.9271±.0207○	.6764±.0361•	.8607±.0304•	.7987±.0259•	.5671±.0446•	.6728±.0261•	.9018±.0262○
Sensor	.9426±.0169	.7174±.0311•	N/A	.8635±.0201•	.9011±.0147•	.6136±.0387•	.6443±.0316•	N/A
Mnist	.8976±.0057	.8985±.0036	N/A	.7251±.0236•	.7324±.0139•	.8618±.0016•	.7046±.0265•	N/A
Fmnist	.8108±.0026	.7247±.0050•	N/A	.6755±.0136•	.6947±.0110•	.4330±.0082•	.2512±.0046•	N/A
Average	.8148±.1306	.7716±.1428	−	.7443±.1582	.7370±.1701	.6465±.1840	.5647±.2201	−
Win/tie/loss		**14/4/2**	**20/0/0**	**20/0/0**	**19/1/0**	**19/0/1**	**19/0/1**	**17/2/1**

6 Conclusion

This work takes one step on the interpretations over feature groups for tree models. We first propose the Baseline Group Shapley value (BGShapvalue) to quantify the importance of a feature group for tree models. We then develop a polynomial algorithm, BGShapTree, to deal with the sum of exponential terms in the BGShapvalue. Based on this approach, we further greedily search salient feature groups to interpret tree models' predictions. This work can be generalized to other tree-based models such as XGBoost and deep forests. It is also interesting to exploit other baseline instances and efficient algorithms to search better salient feature groups. We leave these to future works.

Acknowledgements The authors want to thank the editors and reviewers for their helpful comments and suggestions. The authors also thank Jia-He Yao for helpful advice. This research was supported by the National Science and Technology Major Project (2021ZD0112802) and the National Natural Science Foundation of China (Grant No. 62376119).

Competing interests The authors declare that they have no competing interests or financial conflicts to disclose.

References

1. Breiman L. Random forests. Machine Learning, 2001, 45(1): 5−32
2. 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. 2016, 785−794
3. Zhou Z H, Feng J. Deep forest: towards an alternative to deep neural networks. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. 2017, 3553−3559
4. 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. 2016, 1135−1144
5. Grath R M, Costabello L, Le Van C, Sweeney P, Kamiab F, Shen Z, Lecue F. Interpretable credit application predictions with counterfactual explanations. 2018, arXiv preprint arXiv: 1811.05245
6. Lundberg S M, Nair B, Vavilala M S, Horibe M, Eisses M J, Adams T, Liston D E, Low D K W, Newman S F, Kim J, Lee S I. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. Nature Biomedical Engineering, 2018, 2(10): 749−760
7. Tjoa E, Guan C. A survey on explainable artificial intelligence (XAI): toward medical XAI. IEEE Transactions on Neural Networks and Learning Systems, 2021, 32(11): 4793−4813
8. Zablocki É, Ben-Younes H, Pérez P, Cord M. Explainability of deep vision-based autonomous driving systems: review and challenges. International Journal of Computer Vision, 2022, 130(10): 2425−2452
9. Breiman L, Friedman J, Olshen R A, Stone C J. Classification and Regression Trees. New York: CRC Press, 1984
10. Strobl C, Boulesteix A L, Zeileis A, Hothorn T. Bias in random forest variable importance measures: illustrations, sources and a solution. BMC Bioinformatics, 2007, 8: 25
11. Louppe G, Wehenkel L, Sutera A, Geurts P. Understanding variable importances in forests of randomized trees. In: Proceedings of the 26th International Conference on Neural Information Processing Systems. 2013, 431−439
12. Saabas A. Interpreting random forests. See interpreting-random-forests/ website, 2014
13. Kazemitabar S J, Amini A A, Bloniarz A, Talwalkar A. Variable importance using decision trees. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017, 425−434
14. Li X, Wang Y, Basu S, Kumbier K, Yu B. A debiased MDI feature importance measure for random forests. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019, 723
15. Shapley L S. A value for n-person games. In: Kuhn H W, Tucker A W, eds. Contributions to the Theory of Games. Princeton: Princeton University Press, 1953, 307−317

16. 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, 2020, 2(1): 56−67
17. Athanasiou M, Sfrintzeri K, Zarkogianni K, Thanopoulou A C, Nikita K S. An explainable XGBoost–based approach towards assessing the risk of cardiovascular disease in patients with type 2 diabetes mellitus. In: Proceedings of the 20th IEEE International Conference on Bioinformatics and Bioengineering. 2020, 859−864
18. Feng D C, Wang W J, Mangalathu S, Taciroglu E. Interpretable XGBoost-SHAP machine-learning model for shear strength prediction of squat RC walls. Journal of Structural Engineering, 2021, 147(11): 04021173
19. Sutera A, Louppe G, Huynh-Thu V A, Wehenkel L, Geurts P. From global to local MDI variable importances for random forests and when they are Shapley values. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021, 3533−3543
20. Amoukou S I, Salaün T, Brunel N J B. Accurate Shapley values for explaining tree-based models. In: Proceedings of the 25th International Conference on Artificial Intelligence and Statistics. 2022, 2448−2465
21. Sundararajan M, Najmi A. The many Shapley values for model explanation. In: Proceedings of the 37th International Conference on Machine Learning. 2020, 859
22. Lundberg S M, Lee S I. A unified approach to interpreting model predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017, 4768−4777
23. Marichal J L. The influence of variables on pseudo-Boolean functions with applications to game theory and multicriteria decision making. Discrete Applied Mathematics, 2000, 107(1-3): 139−164
24. Flores R, Molina E, Tejada J. Evaluating groups with the generalized Shapley value. 4OR, 2019, 17(2): 141−172
25. Marichal J L, Kojadinovic I, Fujimoto K. Axiomatic characterizations of generalized values. Discrete Applied Mathematics, 2007, 155(1): 26−43
26. Sundararajan M, Taly A, Yan Q. Axiomatic attribution for deep networks. In: Proceedings of the 34th International Conference on Machine Learning. 2017, 3319−3328
27. Štrumbelj E, Kononenko I. Explaining prediction models and individual predictions with feature contributions. Knowledge and Information Systems, 2014, 41(3): 647−665
28. Datta A, Sen S, Zick Y. Algorithmic transparency via quantitative input influence: theory and experiments with learning systems. In: Proceedings of 2016 IEEE Symposium on Security and Privacy. 2016, 598−617
29. Díaz-Uriarte R, de Andrés S A. Gene selection and classification of microarray data using random forest. BMC Bioinformatics, 2006, 7: 3
30. Ishwaran H. Variable importance in binary regression trees and forests. Electronic Journal of Statistics, 2007, 1: 519−537
31. Archer K J, Kimes R V. Empirical characterization of random forest variable importance measures. Computational Statistics & Data Analysis, 2008, 52(4): 2249−2260
32. Strobl C, Boulesteix A L, Kneib T, Augustin T, Zeileis A. Conditional variable importance for random forests. BMC Bioinformatics, 2008, 9: 307
33. Auret L, Aldrich C. Empirical comparison of tree ensemble variable importance measures. Chemometrics and Intelligent Laboratory Systems, 2011, 105(2): 157−170
34. Louppe G. Understanding random forests: from theory to practice. 2014, arXiv preprint arXiv: 1407.7502
35. Nembrini S, König I R, Wright M N. The revival of the Gini importance?. Bioinformatics, 2018, 34(21): 3711−3718
36. Scornet E. Trees, forests, and impurity-based variable importance. 2020, arXiv preprint arXiv: 2001.04295
37. Sagi O, Rokach L. Explainable decision forest: transforming a decision forest into an interpretable tree. Information Fusion, 2020, 61: 124−138
38. Tan S, Soloviev M, Hooker G, Wells M T. Tree space prototypes: another look at making tree ensembles interpretable. In: Proceedings of 2020 ACM-IMS on Foundations of Data Science Conference. 2020, 23−34
39. Lucic A, Oosterhuis H, Haned H, de Rijke M. FOCUS: flexible optimizable counterfactual explanations for tree ensembles. In: Proceedings of the 36th AAAI Conference on Artificial Intelligence. 2022, 5313−5322
40. Parmentier A, Vidal T. Optimal counterfactual explanations in tree ensembles. In: Proceedings of the 38th International Conference on Machine Learning. 2021, 8422−8431
41. Dutta S, Long J, Mishra S, Tilli C, Magazzeni D. Robust counterfactual explanations for tree-based ensembles. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 5742−5756
42. Ignatiev A. Towards trustable explainable AI. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence. 2020, 5154−5158
43. Izza Y, Ignatiev A, Marques-Silva J. On explaining decision trees. 2020, arXiv preprint arXiv: 2010.11034
44. Izza Y, Marques-Silva J. On explaining random forests with SAT. In: Proceedings of the 30th International Joint Conference on Artificial Intelligence. 2021, 2584−2591
45. Ignatiev A, Izza Y, Stuckey P J, Marques-Silva J. Using MaxSAT for efficient explanations of tree ensembles. In: Proceedings of the 36th AAAI Conference on Artificial Intelligence. 2022, 3776−3785
46. Agarwal A, Tan Y S, Ronen O, Singh C, Yu B. Hierarchical shrinkage: improving the accuracy and interpretability of tree-based models. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 111−135
47. Yang J. Fast TreeSHAP: accelerating SHAP value computation for trees. 2021, arXiv preprint arXiv: 2109.09847
48. Grömping U. Estimators of relative importance in linear regression based on variance decomposition. The American Statistician, 2007, 61(2): 139−147
49. Sun Y, Sundararajan M. Axiomatic attribution for multilinear functions. In: Proceedings of the 12th ACM Conference on Electronic Commerce. 2011, 177−178
50. Aas K, Jullum M, Løland A. Explaining individual predictions when features are dependent: more accurate approximations to Shapley values. Artificial Intelligence, 2021, 298: 103502
51. Chau S L, Hu R, Gonzalez J, Sejdinovic D. RKHS-SHAP: Shapley values for kernel methods. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022, 13050−13063
52. Ancona M, Oztireli C, Gross M. Explaining deep neural networks with a polynomial time algorithm for Shapley value approximation. In: Proceedings of the 36th International Conference on Machine Learning. 2019, 272−281
53. Ghorbani A, Zou J. Neuron Shapley: discovering the responsible neurons. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 5922−5932
54. Bento J, Saleiro P, Cruz A F, Figueiredo M A T, Bizarro P. TimeSHAP: explaining recurrent models through sequence perturbations. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2021, 2565−2573
55. Wang G, Chuang Y N, Du M, Yang F, Zhou Q, Tripathi P, Cai X, Hu X. Accelerating Shapley explanation via contributive cooperator

selection. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 22576−22590
56. Chen L, Lou S, Zhang K, Huang J, Zhang Q. HarsanyiNet: computing accurate Shapley values in a single forward propagation. In: Proceedings of the 40th International Conference on Machine Learning. 2023, 4804−4825
57. Štrumbelj E, Kononenko I, Šikonja M R. Explaining instance classifications with interactions of subsets of feature values. Data & Knowledge Engineering, 2009, 68(10): 886−904
58. Owen A B. Sobol' indices and Shapley value. SIAM/ASA Journal on Uncertainty Quantification, 2014, 2(1): 245−251
59. Owen A B, Prieur C. On Shapley value for measuring importance of dependent inputs. SIAM/ASA Journal on Uncertainty Quantification, 2017, 5(1): 986−1002
60. Frye C, Rowat C, Feige I. Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 1229−1239
61. Heskes T, Sijben E, Bucur I G, Claassen T. Causal Shapley values: exploiting causal knowledge to explain individual predictions of complex models. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 4778−4789
62. Dhamdhere K, Agarwal A, Sundararajan M. The Shapley Taylor interaction index. In: Proceedings of the 37th International Conference on Machine Learning. 2020, 9259−9268
63. Covert I, Lee S I. Improving KernelSHAP: practical Shapley value estimation using linear regression. In: Proceedings of the 24th International Conference on Artificial Intelligence and Statistics. 2021, 3457−3465
64. Janizek J D, Sturmfels P, Lee S I. Explaining explanations: axiomatic feature interactions for deep networks. The Journal of Machine Learning Research, 2021, 22(1): 104
65. Wang J, Zhang Y, Gu Y, Kim T K. SHAQ: incorporating Shapley value theory into multi-agent Q-learning. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022, 5941−5954
66. Beechey D, Smith T M S, Şimşek Ö. Explaining reinforcement learning with Shapley values. In: Proceedings of the 40th International Conference on Machine Learning. 2023, 2003−2014
67. Ren J, Zhang D, Wang Y, Chen L, Zhou Z, Chen Y, Cheng X, Wang X, Zhou M, Shi J, Zhang Q. Towards a unified game-theoretic view of adversarial perturbations and robustness. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021, 3797−3810
68. Chau S L, Muandet K, Sejdinovic D. Explaining the uncertain: stochastic Shapley values for Gaussian process models. In: Proceedings of the 37th Conference on Neural Information Processing Systems. 2023, 50769−50795
69. Watson D S, O'Hara J, Tax N, Mudd R, Guy I. Explaining predictive uncertainty with information theoretic Shapley values. In: Proceedings of the 37th Conference on Neural Information Processing Systems. 2023, 7330−7350
70. Janzing D, Minorics L, Blöbaum P. Feature relevance quantification in explainable AI: a causal problem. In: Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics. 2020, 2907−2916
71. Kumar I E, Venkatasubramanian S, Scheidegger C, Friedler S A. Problems with Shapley-value-based explanations as feature importance measures. In: Proceedings of the 37th International Conference on Machine Learning. 2020, 5491−5500
72. Kumar I E, Scheidegger C, Venkatasubramanian S, Friedler S A. Shapley residuals: quantifying the limits of the Shapley value for explanations. In: Proceedings of the 35th Conference on Neural Information Processing Systems. 2021, 26598−26608
73. Kwon Y, Zou J. WeightedSHAP: analyzing and improving Shapley based feature attributions. In: Proceedings of the 36th Conference on Neural Information Processing Systems. 2022, 34363−34376
74. Van den Broeck G, Lykov A, Schleich M, Suciu D. On the tractability of SHAP explanations. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence. 2021, 6505−6513
75. Bordt S, von Luxburg U. From Shapley values to generalized additive models and back. In: Proceedings of the 26th International Conference on Artificial Intelligence and Statistics. 2023, 709−745
76. Jullum M, Redelmeier A, Aas K. groupShapley: efficient prediction explanation with Shapley values for feature groups. 2021, arXiv preprint arXiv: 2106.12228
77. Miroshnikov A, Kotsiopoulos K, Filom K, Kannan A R. Stability theory of game-theoretic group feature explanations for machine learning models. 2021, arXiv preprint arXiv: 2102.10878
78. Au Q, Herbinger J, Stachl C, Bischl B, Casalicchio G. Grouped feature importance and combined features effect plot. Data Mining and Knowledge Discovery, 2022, 36(4): 1401−1450
79. Vanschoren J, van Rijn J N, Bischl B, Torgo L. OpenML: networked science in machine learning. ACM SIGKDD Explorations Newsletter, 2014, 15(2): 49−60
80. Kelly M, Longjohn R, Nottingham K. The UCI Machine Learning Repository. See archive.ics.uci.edu website. 2024
81. Samek W, Binder A, Montavon G, Lapuschkin S, Müller K R. Evaluating the visualization of what a deep neural network has learned. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(11): 2660−2673
82. Lundberg S M, Erion G G, Lee S I. Consistent individualized feature attribution for tree ensembles. 2018, arXiv preprint arXiv: 1802.03888

Fan Xu received his BSc degree from Southeast University, China in 2020. Currently, he is working towards the PhD degree in Nanjing University, China. His research interest is mainly on machine learning.

Zhi-Jian Zhou received his BSc degree from Dalian University of Technology, China in 2021. He is now a graduate student in Nanjing University, China. His research interest is mainly on hypothesis testing.

Jie Ni received his BSc degree from Nanjing University, China in 2021. Currently, he is a graduate student in Nanjing University, China. His research interest include machine learning and data mining.

Wei Gao received his PhD degree from Nanjing University, China in 2014, and he is currently an associate professor of School of Artificial Intelligence in Nanjing University, China. His research interests include learning theory. His works have been published in top-tier international journals or conference proceedings such as AIJ, IEEE TPAMI, COLT, ICML and NeurIPS. He is also a co-author of the book *Introduction to the Theory of Machine Learning*.