Attention and self-attention in random forests

Lev V. Utkin · Andrei V. Konstantinov · Stanislav R. Kirpichenko

Received: 2 August 2022 / Accepted: 20 March 2023 / Published online: 11 May 2023
© Springer-Verlag GmbH Germany, part of Springer Nature 2023

Abstract
New models of random forests jointly using the attention and self-attention mechanisms are proposed for solving the regression problem. The models can be regarded as extensions of the attention-based random forest whose idea stems from applying a combination of the Nadaraya–Watson kernel regression and the Huber’s contamination model to random forests. The self-attention aims to capture dependencies of the tree predictions and to remove noise or anomalous predictions in the random forest. The self-attention module is trained jointly with the attention module for computing weights. It is shown that the training process of attention weights is reduced to solving a single quadratic or linear optimization problem. Three modifications of the self-attention are proposed and compared. A specific multi-head self-attention for the random forest is also considered. Heads of the self-attention are obtained by changing its tuning parameters including the kernel parameters and the contamination parameter of models. The proposed modifications of the attention and self-attention combinations are verified and compared with other random forest models by using several datasets. The code implementing the corresponding algorithms is publicly available.

Keywords Attention mechanism · Random forest · Nadaraya–Watson regression · Quadratic programming · Linear programming · Contamination model

1 Introduction

The attention mechanism is an effective method for improving the performance of neural networks. It was proposed to enhance the natural language processing models, and later becomes one of the most effective methods for various machine learning tasks. A neural network with attention-based components can automatically distinguish the relative importance of features or examples by means of assigning the corresponding weights to them to get a higher classification or regression accuracy. The main idea behind the attention mechanism stems from the human perception property to concentrate on an important part of information and to ignore other information [40]. Due to the ability of the attention to significantly improve the neural network performance, a huge amount of attention-based models have been developed to be used in various applications and tasks [7, 12, 13, 33, 40].

In spite of success of many neural attention models in solving various application tasks, attention is a component of neural architectures [7]. This implies that the attention weights are learned by incorporating an additional feed-forward neural network within the architectures. The corresponding models meet difficulties of neural networks, including, overfitting, many tuning parameters, requirements of a large amount of data, the black-box nature, expensive computations. Moreover, tabular learning data may be also an important problem encountered with neural networks. There are several deep learning models [1, 27, 51] illustrating efficiency on some tabular datasets. However, other experimental studies [4, 50] give opportunity to conclude that ensemble-based models using decision trees as weak learners mainly outperform deep neural networks when they deal with heterogeneous tabular data.

Taking the above into account, Utkin and Konstantinov [30, 55] proposed a new model called the attention-based random forest (ABRF) by incorporating the attention mecha-
anism into ensemble-based models such as random forests (RF) [6] and the gradient boosting machine [20, 21]. The original RF is a powerful model which consists of a large number of randomly built individual decision trees such that their predictions are combined, for example, by means of the simple averaging. Decision trees are built by the random selection of different subsamples of examples from training data and different subsamples of the feature space. The main idea behind the proposed ABRF models stems from the Nadaraya–Watson kernel regression model [39, 65], but attention weights used in the Nadaraya–Watson regression are assigned to decision trees in a specific way. The weights can be regarded as the attention weights because they are defined by using queries, keys and values concepts in terms of the attention mechanism. In contrast to weights of trees defined in [57, 59], weights in ABRF have trainable parameters and depend on how far an example, which falls into a leaf of a tree, is from examples which fall into the same leaf. The resulting prediction of ABRF is computed as a weighted sum of the tree predictions.

Three modifications of ABRF were studied in [30, 55]. The first modification called $\epsilon$-ABRF uses the Huber’s $\epsilon$-contamination model [26] for defining the attention weights. Each weight consists of two parts: the softmax operation with the tuning coefficient $1 - \epsilon$ and the trainable bias of the softmax weight with coefficient $\epsilon$. An important advantage of $\epsilon$-ABRF is that attention weights of trees linearly depend on trainable parameters. This property leads to solving the standard quadratic optimization problem which is simply solved. $\epsilon$-ABRF avoids using the gradient-based algorithm to compute optimal trainable parameters. Other two ABRF modifications differ from $\epsilon$-ABRF by the additional trainable attention parameters incorporated into the softmax operation. These modifications require to apply the gradient-based algorithms to compute optimal attention parameters.

$\epsilon$-ABRF has demonstrated outperforming results for many real datasets. However, an “unfortunate” selection of a subset of training examples for building a tree can lead to anomalous or incorrect predictions which bias the RF prediction. In order to overcome this disadvantage and following the idea behind $\epsilon$-ABRF, we propose to supplement this model by the self-attention mechanism [60] which aims to capture dependencies of the tree predictions and to remove noise or anomalous predictions in $\epsilon$-ABRF. The proposed model is called SAT-RF (self-attention-attention-based RF). The main peculiarity of the supplemented self-attention is that it is trained jointly with the attention mechanism, but not sequentially, i.e., we solve a single optimization problem for simultaneous computing trainable parameters of $\epsilon$-ABRF and the supplemented self-attention. As a result, both the mechanisms impact each other and can be regarded as a joint attention-based modification. The use of the Huber’s $\epsilon$-contamination model with parameter $\epsilon$ different from $\epsilon$-ABRF for defining the self-attention leads to the quadratic or linear optimization problem with trainable parameters of $\epsilon$-ABRF and the supplemented self-attention as optimization variables. This is an important property of the proposed model. Moreover, we proposed a specific variant of the multi-head self-attention which allows us to combine knowledge of the self-attention via different representation of its tuning parameters. It is shown that the multi-head self-attention also leads to the quadratic or linear optimization problem for computing trainable parameters of all heads.

Our contributions can be summarized as follows:

1. A new attention-based RF model is proposed. According to the model, the trainable self-attention mechanism is incorporated into the attention-based RF as an additional component to capture dependencies of the tree predictions and to remove noise or anomalous predictions in $\epsilon$-ABRF. It is important that the self-attention and attention components are jointly trained such that trainable parameters of the attention impact on parameters of the self-attention and vice versa.

2. Three modifications of the self-attention are studied. The first one (SAT-RF-y) is based on comparison of predictions provided by pairs of trees. The second modification (SAT-RF-x) takes into account only distances between mean feature vectors which are determined from all feature vectors which fall into the same leaves with the tested example in pairs of trees. The third modification (SAT-RF-yx) can be regarded as a combination of the first and the second modifications.

3. A specific multi-head self-attention for the RF is proposed. Heads are obtained by changing the tuning parameters of the self-attention. They are trained by solving a single quadratic optimization problem for computing the optimal attention and self-attention weights. It can be said that the whole model is trained “end-to-end” to some extent.

4. Various numerical experiments with real tabular datasets are provided to justify SAT-RF, to study its peculiarities and to compare it with original RFs. Moreover, we investigate two types of RFs: original RFs and extremely randomized trees (ERT). At each node, the ERT algorithm chooses a split point randomly for each feature and then selects the best split among these [22]. The corresponding code implementing SAT-RF is publicly available.

The paper is organized as follows. Related work can be found in Sect. 2. A brief introduction to the attention and self-attention mechanisms is given in Sect. 3. A general approach to incorporating the attention and the self-attention into the RF is provided in Sect. 4. Analysis of the attention and

---

1 https://github.com/andruekonst/forest-self-attention.
self-attention representations by means of the Huber’s \( \epsilon \)-contamination model is given in Sect. 5. Some questions of applying the multi-head self-attention in the framework of the general approach is considered in Sect. 6. Numerical experiments with real data illustrating the efficiency of the proposed models for solving the regression problems are provided in Sect. 7. Concluding remarks can be found in Sect. 8.

2 Related work

2.1 Attention mechanism and approaches for its simplification

Many variants of the attention mechanism have been developed and studied since its first proposals [2, 37, 60]. Their comprehensive description and analysis can be found in many survey papers [7, 12, 13, 33, 35, 40].

It is difficult to overestimate the contribution of the attention mechanism into many deep learning models. At the same time, one of the obstacles of the attention as well as self-attention mechanisms is the quadratic time of training due to the softmax function which is used for computing the attention score function. This computational problem was solved by several approaches. One of the approaches is to utilize the sparse attention [3, 36, 42]. Another important approach is the kernel linearization. In the framework of this approach, a specific transformer architecture called Performer was introduced by Choromanski et al. [11]. Due to the linearization, performer estimates softmax attention using only linear time complexity. Another approximation of the softmax attention called Luna (a linear unified nested attention mechanism) was proposed by Ma et al. [38]. Luna approximates softmax attention with two nested linear attention functions. Choromanski et al. [10] introduced a new class of random feature maps techniques called hybrid random features which provide more accurate worst-case softmax kernel estimation than previous algorithms and lead to computational gains. Peng et al. [43] solved the same problem and proposed the random feature attention method which also approximates the softmax function and can be used as a drop-in replacement for the conventional softmax attention. Schlag et al. [47] first listed desirable properties of a kernel function, which is the central component of softmax linearization, reviewed the existing functions and proposed a new function having these properties. In order to linearize the exponential kernel and to approximate the softmax attention in linear time and space complexity, Zheng et al. [75] proposed a method based on recasting random feature attentions as self-normalized importance samplers. Detailed reviews on approaches related to problems of the efficient attention mechanism implementation can be found in [19, 33, 35, 53].

2.2 Self-attention

The self-attention was proposed by Vaswani et al. [60] as an important component of a new neural network architecture known as transformer. It is inspired by the previous works presented by Cheng et al. [9], where self-attention is called intra-attention, by Parikh et al. [41]. The self-attention aims to capture token dependencies and to relate distinct positions in the input sequence. It has been used in many tasks, for example, sentence embedding [34], in machine translation and natural language processing [15, 17, 67], in speech recognition [44, 49, 62], in image recognition [8, 24, 28, 45, 48, 63, 64, 74].

Many survey papers are devoted to various aspects and applications of attention and self-attention mechanisms, for example, [5, 7, 23, 25, 28, 33, 52, 68].

We use self-attention to remove anomalies in the tree predictions. Similar approaches to image denoising were considered in [32, 54, 61, 70, 77].

It should be noted that the above methods are implemented as a part of a neural network, and they are not studied for application to other machine learning models, for example, to RFs.

2.3 Weighted RFs

An idea of assigning or incorporating weights to trees in RFs can be regarded as one of the approaches to improve the classification and regression performance of the RF models. As a result, several models were proposed to incorporate weights of trees into RFs and to combine decision trees taking into account the weights. We can conditionally divide the proposed weighted RF models into two classes. The first class considers weights as tuning parameters. In fact, weights in models from the first class are assigned by using some prior information about training data, for example, the data class imbalance, or posterior information obtained in the model validation stage. Many models from the first class can be found in [14, 29, 31, 46, 66, 69, 73].

In contrast to the models with weights as hyperparameters, the second class considers weights of trees as training parameters. In the models, weights are computed by solving optimization problems in accordance with certain loss functions which aim to minimize the RF classification and regression error [56, 58] and to robustify whole models for constructing cautious RFs [57, 59, 72]. Another approach, which can be regarded as an attempt to incorporate the attention mechanism into RFs and the gradient boosting machine, was proposed in [30, 55]. It is a basis for developing new models using the attention mechanism and its extensions.
3 Preliminaries

3.1 Attention mechanism and the Nadaraya–Watson regression

There are many views on attention mechanisms. Most of them consider attention mechanisms as components of a neural network, which help to focus on relevant regions of input data to improve the network classification performance. However, one of the views [7, 71] is to understand the attention by considering the well-known Nadaraya–Watson kernel regression model [39, 65]. The Nadaraya–Watson kernel regression allows us to understand the statistical nature of attention and to apply this representation for RFs.

First, we formulate the standard regression task. Let us consider a dataset $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ consisting of $n$ examples such that $x_i = (x_{i1}, \ldots, x_{im}) \in \mathbb{R}^m$ is a vector of $m$ features and $y_i \in \mathbb{R}$ represents the regression output. The regression task is to construct a function $f : \mathbb{R}^m \rightarrow \mathbb{R}$ which accurately predicts the output value $y$ of a new example $x$ based on dataset $S$. The classification task can be formulated in the same way under condition that $y_i$ takes values from a finite set.

According to the Nadaraya–Watson regression model [39, 65], the output value or prediction $\hat{y}$ can be computed by means of the weighted average as follows:

$$\hat{y} = \sum_{i=1}^{n} \alpha(x, x_i) y_i. \tag{1}$$

Here weight $\alpha(x, x_i)$ is nothing but the attention weight which conforms with relevance of the $i$-th example to the vector $x$. In other words, weight $\alpha(x, x_i)$ is assigned to every $y_i$ from the dataset depending on location of vector $x_i$ relative to the predicted vector $x$. The closer $x_i$ to $x$, the greater the weight assigned to $y_i$ corresponding to $x_i$. The last property of weights motivates to represent them through a kernel $K(x, x_i)$, i.e., we can write the following definition for weights:

$$\alpha(x, x_i) = \frac{K(x, x_i)}{\sum_{j=1}^{n} K(x, x_j)}. \tag{2}$$

Vectors $x, x_i$ and outputs $y_i$ are called the query, keys and values, respectively [2].

It is important to point out that if we use the Gaussian kernel, then weights are expressed through the softmax operator with a hyperparameter $\tau$ as:

$$\alpha(x, x_i) = \text{softmax} \left(-\frac{\|x - x_i\|^2}{2\tau} \right). \tag{3}$$

3.2 Self-attention mechanism as the non-local means denoising

One of the interesting interpretations of the self-attention mechanism is the non-local means denoising [61], which aims to remove noise in an image by computing average intensity of each pixel from a set of neighboring pixels. This idea again stems from the Nadaraya–Watson regression under condition that the query is a key, and each key coincides with the corresponding value.

According to [61], intensity of a pixel with coordinates $x$ by using the non-local means denoising is determined as follows:

$$f(x) = y^s = \sum_{j=1}^{n} \beta(y, y_j) y_j, \tag{4}$$

where weight $\beta(y, y_j)$ is determined as

$$\beta(y, y_j) = \frac{K(y, y_j)}{\sum_{j=1}^{n} K(y, y_j)}. \tag{5}$$

If to use the Gaussian kernel, then weights are of the form:

$$\beta(y, y_j) = \text{softmax} \left(-\frac{(y - y_j)^2}{2\kappa} \right), \tag{6}$$

where $\kappa$ is the tuning or training parameter.

SAT-RF with the above definition of the softmax operation is called SAT-RF-y. Generally, the query $f(x)$ and values $y_j$ can be vectors. Moreover, the values can be taken in another form. Variants of the forms are considered below.

4 Self-attention-based random forest

One of the powerful machine learning models handling with tabular data is the RF which can be regarded as an ensemble of $T$ decision trees such that each tree is trained on a subset of examples randomly selected from the training set. In the original RF, the final RF prediction $\hat{y}$ for a testing example $x$ is determined by averaging predictions $\hat{y}_1, \ldots, \hat{y}_T$ obtained for all trees.

Denote an index set of examples which fall into the $i$-th leaf in the $k$-th tree as $\mathcal{J}_i^{(k)}$ such that $\mathcal{J}_i^{(k)} \cap \mathcal{J}_j^{(k)} = \emptyset$ because the same example cannot fall into different leaves of the same tree. Let us consider an example $x$ which falls into $i$-th leaf in the $k$-th tree. Then we can introduce the mean vector $A_k(x)$ defined as the mean of training vectors $x_j$ which fall into the $i$-th leaf of the $k$-th tree, i.e., into the leaf where vector $x$ fell into. In the same way, we introduce the mean target value $B_k(x)$ defined as the mean of $y_j$ such that $j \in \mathcal{J}_i^{(k)}$. In fact,
value $B_k(x)$ in regression coincides with the prediction of the $k$-th tree. Formally, we write

$$A_k(x) = \frac{1}{|J_k(x)|} \sum_{j \in J_k(x)} x_j,$$

$$B_k(x) = \frac{1}{|J_k(x)|} \sum_{j \in J_k(x)} y_j.$$  \hfill (7)  

By returning to the Nadaraya–Watson regression and notation of the attention mechanism framework, the set of $A_k(x)$, $k = 1, ..., T$, can be regarded as a set of keys for every $x$, the set of $B_k(x)$ can be regarded as a set of values. This implies that the final prediction $\tilde{y}$ of the RF can be computed by using the Nadaraya–Watson regression, namely,

$$\tilde{y} = f(x, w) = \sum_{k=1}^{T} \alpha(x, A_k(x), w_k) \cdot B_k(x).$$  \hfill (9)  

Here $\alpha(x, A_k(x), w_k)$ is the attention weight with vector $w = (w_1, ..., w_T)$ of trainable parameters assigned to the $k$-th tree. One can see that the set of parameters $\theta$ is replaced with $w$. If $\alpha$ is the normalized kernel, then it is defined through the distance between $x$ and $A_k(x)$, which is defined, for instance, by means of $L_2$-norm $\|x - A_k(x)\|^2$. It is assumed that

$$\sum_{k=1}^{T} \alpha(x, A_k(x), w_k) = 1,$$  \hfill (10)  

$$\sum_{k=1}^{T} w_k = 1.$$  \hfill (11)  

Condition (10) is due to properties of the attention weights in the Nadaraya–Watson regression. Condition (11) is explained below when the Huber’s $\epsilon$-contamination model will be considered for representing the attention weights.

The above approach to incorporating the attention mechanism into the RF has been proposed in [30, 55]. Our aim now is to supplement it with the self-attention.

We suppose that there may be anomalies among values $B_k(x)$ or $\tilde{y}_k$. In order to cope with the anomalies, we apply the self-attention mechanism which corrects every $\tilde{y}_k$. According to the self-attention, each $\tilde{y}_k$ can be recalculated as follows:

$$y^*_j = \sum_{i=1}^{T} \beta(\tilde{y}_i, \tilde{y}_j, v_i) \cdot \tilde{y}_i.$$  \hfill (12)  

Here $\beta(\tilde{y}_i, \tilde{y}_j, v_k)$ is the self-attention weight with vector $v = (v_1, ..., v_T)$ of trainable parameters assigned to the $k$-th tree such that $\sum_{k=1}^{T} v_k = 1$.

The main idea behind the approach is to use the attention and self-attention simultaneously. Let us substitute (12) into (9) under condition $B_k(x) = \tilde{y}_k$ as

$$\tilde{y} = f(x, w, v) = \sum_{i=1}^{T} \sum_{k=1}^{T} \alpha(x, A_i(x), w_i) \times \beta(\tilde{y}_i, \tilde{y}_k, v_k) \cdot \tilde{y}_k.$$  \hfill (13)  

We get the trainable attention-based RF with parameters $w$ and $v$, which are defined by minimizing the expected loss function over set $\mathcal{W}$ and set $\mathcal{V}$ of parameters, respectively, as follows:

$$(w_{opt}, v_{opt}) = \arg \min_{w \in \mathcal{W}, v \in \mathcal{V}} \sum_{s=1}^{n} L(\tilde{y}_s, y_s, w, v).$$  \hfill (14)  

The loss function can be rewritten as

$$\sum_{s=1}^{n} L(\tilde{y}_s, y_s, w, v) = \sum_{s=1}^{n} \left( y_s - \sum_{i=1}^{T} \alpha_i \cdot \beta_k \cdot \tilde{y}_k \right)^2.$$  \hfill (15)  

where

$$\alpha_i = \alpha(x, A_i(x), w_i), \beta_k = \beta(\tilde{y}_i, \tilde{y}_k, v_k).$$

Optimal trainable parameters $w, v$ are computed depending on forms of attention weights $\alpha$ and self-attention weights $\beta$. Moreover, the computation time for solving the optimization problem (15) also significantly depends on the weights. Therefore, we propose the form which leads to convex quadratic optimization problem.

It can be seen from the above that every value $y$ is transformed to $y^*$ in accordance with the difference between $y$ and other values $y_i$. However, the above non-local means denoising does not take into account the distance between the vectors $A_i(x)$ and $A_j(x)$. In other words, it is interesting to take into account how the mean feature vector of all feature vectors which fall into the same leaves with $x$ of the $i$-th and the $j$-th trees, respectively. Hence, we can write the self-attention weight as

$$\beta(A_i, A_j) = \text{softmax} \left( -\frac{\|A_i(x) - A_j(x)\|^2}{2\kappa} \right).$$  \hfill (16)  

SAT-RF with the above definition of the softmax operation is called SAT-RF-x.
By intuition, if \( A_i(x) \) and \( A_j(x) \) are close to each other, then we can expect that the difference between values \( y_i \) and \( y_j \) is small. If it is large, then the weight of \( y_i \) should be larger than in the case when the difference between values \( y_i \) and \( y_j \) is small. On the contrary, if \( A_i(x) \) and \( A_j(x) \) are far from each other, then the impact of value \( y_j \) is reduced and the corresponding weight should be decreased even if the difference between values \( y_i \) and \( y_j \) is small. The above reasoning leads to applying the following self-attention weights:

\[
\beta(y_i, y_j) = \text{softmax} \left( -\frac{(y_i - y_j)^2}{2\kappa \| A_i(x) - A_j(x) \|^2} \right),
\]

(17)

SAT-RF with the above definition of the softmax operation is called SAT-RF-yx.

It should be pointed out that the modifications of SAT-RF do not impact on the general approach, and they define only the softmax operations. Therefore, all expressions will be given using the first modifications, but results of numerical experiments will be considered for every modification.

5 Self-attention and the Huber’s contamination model

To simplify computations and to get a unique solution for \( w \), we propose to use the well-known Huber’s \( \epsilon \)-contamination model [26] which can be represented as follows:

\[
(1 - \epsilon) \cdot P + \epsilon \cdot Q,
\]

where the probability distribution \( P \) is contaminated by some arbitrary distribution \( Q \); the rate \( \epsilon \in [0, 1] \) is a model parameter which control the size of the solution set.

The use of the \( \epsilon \)-contamination model stems from several reasons. First of all, the softmax function can be interpreted as the probability distribution \( P \) in (18) because its sum is 1. It can be represented as a point in the probabilistic unit simplex having \( T \) vertices. Second, weights \( \alpha(x, A_i(x), w_i) \) also can be interpreted as a probability distribution or another point in the same unit simplex. This point is biased by means of the probability distribution \( Q \) in (18) which is trained in order to achieve the best prediction results. The contamination parameter \( \epsilon \) can be regarded as a tuning parameter of the model. It should be noted that \( \epsilon \) can be viewed as the trainable parameter. However, this case leads to a more complex optimization problem. After substituting elements of \( \alpha(x, A_i(x), w_i) \) into (18), we get

\[
\alpha(x, A_i(x), w_i) = (1 - \epsilon) \cdot \text{softmax} \left( \frac{\| x_i - A_i(x_i) \|^2}{\tau} \right) + \epsilon \cdot w_i.
\]

(19)

Let us define the self-attention weights \( \beta(\tilde{y}_i, \tilde{y}_k, v_k) \) in the same way using the Huber’s \( \gamma \)-contamination model. In this case, we can write the similar expression:

\[
\beta(\tilde{y}_i, \tilde{y}_k, v_k) = (1 - \gamma) \cdot \text{softmax} \left( \frac{(\tilde{y}_i - \tilde{y}_k)^2}{\kappa} \right) + \gamma \cdot v_k.
\]

(20)

Here \( \gamma \) is the same parameter of the contamination model as \( \epsilon \).

After substituting (19) and (20) into (13), we get

\[
\sum_{i=1}^{T} \sum_{k=1}^{T} (D_{si} + \epsilon w_i) \cdot (C_{ik} + \gamma v_k) \cdot \tilde{y}_k,
\]

(21)

where

\[
D_{si} = (1 - \epsilon) \cdot \text{softmax} \left( \frac{\| x_s - A_i(x_s) \|^2}{\tau} \right),
\]

(22)

\[
C_{ik} = (1 - \gamma) \cdot \text{softmax} \left( \frac{(\tilde{y}_i - \tilde{y}_k)^2}{\kappa} \right).
\]

(23)

Expression (21) can be rewritten as

\[
\sum_{i=1}^{T} \sum_{k=1}^{T} \tilde{y}_k(D_{si}C_{ik} + C_{ik}\epsilon w_i + D_{si}\gamma v_k + \epsilon \gamma w_i v_k)
\]

\[
= \sum_{i=1}^{T} \sum_{k=1}^{T} \tilde{y}_k D_{si} C_{ik} + \epsilon \sum_{i=1}^{T} \sum_{k=1}^{T} \tilde{y}_k C_{ik} w_i
\]

\[
+ \gamma \sum_{i=1}^{T} D_{si} \sum_{k=1}^{T} \tilde{y}_k v_k + \epsilon \gamma \sum_{k=1}^{T} \tilde{y}_k v_k
\]

\[
= R_s + \sum_{i=1}^{T} H_i w_i + \sum_{k=1}^{T} G_{sk} v_k.
\]

(24)

where

\[
R_s = \sum_{i=1}^{T} \sum_{k=1}^{T} \tilde{y}_k D_{si} C_{ik},
\]

\[
G_{sk} = \gamma \left( \sum_{i=1}^{T} D_{si} + \epsilon \right) \tilde{y}_k,
\]

\[
H_i = \epsilon \sum_{k=1}^{T} \tilde{y}_k C_{ik}.
\]

(25)

Notations \( D_{sk}, C_{ik}, R_s, G_{sk}, H_i \) do not depend on \( w \) and \( v \) and are introduced for short.
It follows from (24) that the optimization problem (15) is represented as

\[
\min_{w, v} \sum_{s=1}^{n} L(\tilde{y}_s, y_s, w, v) = \sum_{s=1}^{n} \left( y_s - R_s - \sum_{i=1}^{T} H_i w_i - \sum_{k=1}^{T} G_{sk} v_k \right)^2,
\]

subject to \( w_k \geq 0, v_k \geq 0, k = 1, \ldots, T, \) and \( \sum_{k=1}^{T} w_k = 1, \sum_{k=1}^{T} v_k = 1. \)

One of the advantages of the proposed SAT-RF is that it is simple from the computational point of view because problem (26) is the standard quadratic programming problem which can be simply solved. Moreover, it has a unique solution.

The optimal trainable parameters \( w \) and \( v \) can be also computed by solving the linear optimization problem if to use the \( L_1 \)-norm for defining the loss function \( L(\tilde{y}_s, y_s, w) \). In this case, we replace (26) with the following objective function:

\[
\min_{w, v} \sum_{s=1}^{n} L(\tilde{y}_s, y_s, w, v) = \sum_{s=1}^{n} \left| y_s - R_s - \sum_{i=1}^{T} H_i w_i - \sum_{k=1}^{T} G_{sk} v_k \right|.
\]

Denote

\[
Q_s = y_s - R_s - \sum_{i=1}^{T} H_i w_i - \sum_{k=1}^{T} G_{sk} v_k.
\]

Then we can write the following linear optimization problem with variables \( Q_1, \ldots, Q_T, w \) and \( v \):

\[
\min_{w, v} \sum_{s=1}^{n} Q_s,
\]

subject to \( w_k \geq 0, v_k \geq 0, k = 1, \ldots, T, \) and \( \sum_{k=1}^{T} w_k = 1, \sum_{k=1}^{T} v_k = 1, \) and for all \( s = 1, \ldots, n, \)

\[
Q_s + \sum_{i=1}^{T} H_i w_i + \sum_{k=1}^{T} G_{sk} v_k \geq y_s - R_s,
\]

\[
Q_s - \sum_{i=1}^{T} H_i w_i - \sum_{k=1}^{T} G_{sk} v_k \geq -y_s + R_s.
\]

The above linear optimization problem has \( 3T \) variables and \( 2T + 3n + 2 \) constraints.

6 Multi-head self-attention

One of the possible extensions of the self-attention mechanism is the multi-head self-attention which is widely used to combine knowledge of the self-attention via different representation of its tuning parameters. It turns out that the multi-head self-attention can be incorporated into the attention-based RF such that its trainable parameters are computed jointly with parameters of the attention-based RF by solving a single quadratic optimization problem.

Let us return to (13) and rewrite the expression for estimating \( \tilde{y} \) as follows:

\[
\tilde{y} = \sum_{i=1}^{T} \sum_{k(1)=1}^{T} \alpha(x, A_i(x), w_i) \beta_1(\tilde{y}_t, \tilde{y}_{k(1)}, v_{k(1)}^{(1)}) \cdot \tilde{y}_{k(1)}, \quad (32)
\]

where \( v^{(1)}(1) = (v_{1}^{(1)}, \ldots, v_{T}^{(1)}) \) is the vector of trainable variable of the first self-attention; \( k(1) \) is the index corresponding to the first self-attention.

Note that \( \tilde{y}_{k(1)} \) as the value in terms of the attention mechanism can be represented by means of the self-attention (12). Hence, (32) can be rewritten as

\[
\tilde{y} = \sum_{i=1}^{T} \sum_{k(1)=1}^{T} \sum_{k(2)=1}^{T} \alpha(x, A_i(x), w_i) \beta_1(\tilde{y}_t, \tilde{y}_{k(1)}, v_{k(1)}^{(1)}) \beta_2(\tilde{y}_t, \tilde{y}_{k(2)}, v_{k(2)}^{(2)}) \cdot \tilde{y}_{k(2)}, \quad (33)
\]

where \( v^{(2)}(2) = (v_{1}^{(2)}, \ldots, v_{T}^{(2)}) \) is the vector of trainable variable of the second self-attention.

In the same way, we can continue writing self-attention operations and get

\[
\tilde{y} = \sum_{i=1}^{T} \sum_{k(1)=1}^{T} \sum_{k(2)=1}^{T} \cdots \sum_{k(t)=1}^{T} \alpha(x, A_i(x), w_i) \beta_1(\tilde{y}_t, \tilde{y}_{k(1)}, v_{k(1)}^{(1)}) \beta_2(\tilde{y}_t, \tilde{y}_{k(2)}, v_{k(2)}^{(2)}) \cdots \beta_t(\tilde{y}_{k(t-1)}, \tilde{y}_{k(t)}, v_{k(t)}^{(t)}) \cdot \tilde{y}_{k(t)}, \quad (34)
\]

In sum, we get a regression with \( t \) self-attention operations having \( t \) self-attention weights \( \beta_1, \ldots, \beta_t \) with \( t \) vectors of trainable parameters \( v^{(1)}, \ldots, v^{(t)} \) and the parameters \( w \) of the attention.

Let us consider the case when the \( j \)-th self-attention weight \( \beta_j(\tilde{y}_{k(j-1)}, \tilde{y}_{k(j)}, v_{k(j)}^{(j)}) \) is represented by the Huber’s
\( \gamma_j \)-contamination model as

\[
\beta_j \left( \tilde{y}_k(j-1), \tilde{y}_k(j), \gamma^j_{k(j)} \right) \\
= (1 - \gamma_j) \times \text{softmax} \left( \frac{(\tilde{y}_k(j-1) - \tilde{y}_k(j))^2}{\kappa_j} \right) \\
+ \gamma_j \cdot \gamma^j_{k(j)}. \tag{35}
\]

Here \( \gamma_j \) and \( \kappa_j \) are tuning parameters of the \( j \)-th contamination model. If parameters \( \gamma_j \) and \( \kappa_j \) are differently defined for different \( j = 1, \ldots, t \), then the obtained scheme can be regarded as an analogue of the original multi-head self-attention. The random choice of values of \( \gamma_j \) and \( \kappa_j \) is similar to the random choice of initial weights in the neural network implementation of the multi-head self-attention.

**Proposition 1** If the self-attention weights \( \alpha(x, A_s(x), w_t) \) and \( \beta_j \left( \tilde{y}_k(j-1), \tilde{y}_k(j), \gamma^j_{k(j)} \right) \) for all \( j = 1, \ldots, t \), are defined by (19) and (35), then \( \tilde{y} \) in (34) is a linear function of parameters \( w, v^{(1)}, \ldots, v^{(t)} \).

**Proof** Introduce the following notations for short:

\[
C_{k(t)} = (1 - \gamma_j) \cdot \text{softmax} \left( \frac{(\tilde{y}_k(j-1) - \tilde{y}_k(j))^2}{\kappa_j} \right). \tag{36}
\]

Then we write

\[
\tilde{y} = \sum_{i=1}^{T} (D_{si} + \epsilon w_t) \sum_{k(1)=1}^{T} \left( C_{k(1)} + \gamma_1 v^{(1)}_{k(1)} \right) \\
\cdots \sum_{k(t-1)=1}^{T} \left( C_{k(t-1)} + \gamma_{t-1} v^{(t-1)}_{k(t-1)} \right) \\
\times \sum_{k(t)=1}^{T} \left( C_{k(t)} + \gamma_t v^{(t)}_{k(t)} \right) \tilde{y}_k(t). \tag{37}
\]

Let us consider \( \beta_{t-1} \) and \( \beta_t \)

\[
\sum_{k(t-1)=1}^{T} \left( C_{k(t-1)} + \gamma_{t-1} v^{(t-1)}_{k(t-1)} \right) \\
\times \sum_{k(t)=1}^{T} \left( C_{k(t)} + \gamma_t v^{(t)}_{k(t)} \right) \tilde{y}_k(t) \\
= \sum_{k(t-1)=1}^{T} \sum_{k(t)=1}^{T} \left( C_{k(t-1)} + \gamma_{t-1} v^{(t-1)}_{k(t-1)} \right) \\
\times \left( C_{k(t)} + \gamma_t v^{(t)}_{k(t)} \right) \tilde{y}_k(t) \\
= \sum_{k(t)=1}^{T} \sum_{k(t-1)=1}^{T} C_{k(t-1)} C_{k(t)} \\
+ \gamma_{t-1} \sum_{k(t-1)=1}^{T} C_{k(t-1)} v^{(t-1)}_{k(t-1)} \\
+ \gamma_t \sum_{k(t-1)=1}^{T} v^{(t)}_{k(t-1)} C_{k(t)} \\
+ \gamma_{t-1} \gamma_t \sum_{k(t-1)=1}^{T} v^{(t-1)}_{k(t-1)} C_{k(t)} \\
+ \gamma_{t-1} \gamma_t \sum_{k(t-1)=1}^{T} v^{(t)}_{k(t-1)} C_{k(t)} \tag{38}
\]

It should be noted that \( \sum_{k(t-1)=1}^{T} v^{(t-1)}_{k(t-1)} = 1 \). This implies that the product of \( \beta_{t-1} \) and \( \beta_t \) linearly depends on \( v^{(t-1)}_{k(t-1)} \) and \( v^{(t)}_{k(t-1)} \). Let us rewrite the obtained product as follows:

\[
\sum_{k(t-1)=1}^{T} \sum_{k(t)=1}^{T} \tilde{y}_k(t) C_{k(t-1)} C_{k(t)} \\
+ \gamma_{t-1} \sum_{k(t-1)=1}^{T} \tilde{y}_k(t) v^{(t-1)}_{k(t-1)} C_{k(t)} \\
+ \gamma_t \sum_{k(t-1)=1}^{T} \tilde{y}_k(t) v^{(t)}_{k(t-1)} C_{k(t)} \\
+ \gamma_{t-1} \gamma_t \sum_{k(t-1)=1}^{T} \tilde{y}_k(t) v^{(t-1)}_{k(t-1)} v^{(t)}_{k(t-1)} \tag{39}
\]

where

\[
G_{k(t)} = \sum_{k(t)=1}^{T} \tilde{y}_k(t) C_{k(t-1)} C_{k(t)}. \tag{40}
\]

is a constant, and

\[
r^{(t)}_{k(t-1)} = v^{(t-1)}_{k(t-1)} \gamma_{t-1} \sum_{k(t-1)=1}^{T} \tilde{y}_k(t) C_{k(t)} \\
+ \gamma_t \sum_{k(t-1)=1}^{T} \tilde{y}_k(t) v^{(t)}_{k(t-1)} C_{k(t)} \\
+ \gamma_{t-1} \gamma_t \sum_{k(t-1)=1}^{T} \tilde{y}_k(t) v^{(t-1)}_{k(t-1)} v^{(t)}_{k(t-1)} \tag{41}
\]

is a new variable instead of \( v^{(t-1)}_{k(t-1)} \) and \( v^{(t)}_{k(t-1)} \). In the same way, we can consider all other multipliers in (37) starting from pair \( \beta_{t-2} \) and \( \beta_{t-1} \), as was to be proved. \( \Box \)

The linearity of \( \tilde{y} \) as the function of variables \( w, v^{(1)}, \ldots, v^{(t)} \) implies that the optimization problem for train-
ing these variables can be reduced to the quadratic or linear optimization problems like (26) or (29). However, writing the optimization problem by \( t > 2 \) is a hard problem. Therefore, the multi-head self-attention was given to show the fundamental possibility of generalizing the self-attention-based RF. An approach for efficient representation of the multi-head attention can be regarded as a separate problem whose solution is a direction for further research.

### 7 Numerical experiments

In order to study the proposed approach for solving regression problems, we apply datasets which are taken from open sources: the dataset Diabetes is available in the corresponding R Packages; datasets Friedman 1, 2 3 can be found at the site\(^2\); Regression and Sparse datasets are available in package “Scikit-Learn”. The proposed algorithm is evaluated and investigated also by the following publicly available datasets from the UCI Machine Learning Repository [18]: Wine Red, Boston Housing, Concrete, Yacht Hydrodynamics, Airfoil. A brief introduction about these data sets are given in Table 1 where \( m \) and \( n \) are numbers of features and examples, respectively. A more detailed information can be found from the aforementioned data resources.

| Dataset                | Abbreviation | \( m \) | \( n \) |
|------------------------|--------------|---------|---------|
| Diabetes               | Diabetes     | 10      | 442     |
| Friedman 1             | Friedman 1   | 10      | 100     |
| Friedman 2             | Friedman 2   | 4       | 100     |
| Friedman 3             | Friedman 3   | 4       | 100     |
| Scikit-learn regression| regression   | 100     | 100     |
| Scikit-learn sparse uncorrelated| sparse | 10      | 100     |
| UCI wine red           | wine         | 11      | 1599    |
| UCI Boston housing     | Boston       | 13      | 506     |
| UCI concrete           | concrete     | 8       | 1030    |
| UCI Yacht hydrodynamics| Yacht        | 6       | 308     |
| UCI airfoil            | airfoil      | 5       | 1503    |

1. **RF, ERT** the original RF or the ERT without the softmax and without attention model;

2. **Softmax model** the RF or the ERT with softmax operations without trainable parameters, i.e., weights of trees are determined under conditions \( \epsilon = 0 \) and \( \gamma = 0 \).

3. **SAT-RF-y, SAT-RF-x, SAT-RF-yx** the corresponding modifications of the SAT-RF models.

Identical best results are shown in bold twice. Moreover, the optimal values of the contamination parameters \( \epsilon_{\text{opt}} \) and \( \gamma_{\text{opt}} \) are provided. The case \( \epsilon_{\text{opt}} = 1 \) means that weights of trees are totally determined by the tree results and do not depend on each example. This case coincides with the weighted RF proposed in [58]. The case \( \gamma_{\text{opt}} = 0 \) means that weights of trees are determined only by the softmax function (with or without trainable parameters). Cases \( \gamma_{\text{opt}} = 1 \) and \( \gamma_{\text{opt}} = 0 \) have the same sense.

Decision trees in numerical experiments are trained such that at least 10 examples fall into every leaf of trees. This condition is used to get desirable estimates of vectors \( A_\mathcal{X}_i (X_i) \).

Every RF or ERT consists of 100 decision trees. In order to optimize the model parameters \( \epsilon \) and \( \tau \) in numerical experiments, we perform a threefold cross-validation on the training set which consists of \( n_{tr} = 4n/5 \) instances. The cross-validation is performed with 100 repetitions. This procedure is realized by considering all possible values of \( \epsilon \) and \( \tau \) in a predefined grid. The testing set consisting of \( n_{\text{test}} = n/5 \) instances is used for computing the accuracy measures of the whole model.

First, we consider SAT-RF-y. It has four tuning parameters \( \epsilon, \gamma, \kappa \) and \( \tau \), which may significantly impact on predictions. Therefore, the best predictions are calculated at a predefined grid of the parameters \( \epsilon, \gamma \), and a cross-validation procedure is subsequently used to select an appropriate values of \( \epsilon \) and \( \gamma \). Moreover, parameters \( \kappa \) and \( \tau \) are taken equal to 1. If SAT-RF-y provides outperforming results with fixed \( \kappa \) and \( \tau \), then optimal choice of these parameters will improve the model.

Measures \( R^2 \) and MAE for three models (RF, Softmax and SAT-RF-y) are shown in Table 2. The results are obtained by training the RF and the parameter vectors \( w \) and \( v \) on the regression datasets. It can be seen from Table 2 that SAT-RF-y outperforms the RF and the Softmax models almost for all datasets. The same results are shown in Table 3 under condition that the RF in experiments is replaced with the ERT. One can again see from Table 3 that SAT-RF-y outperforms the ERT and the Softmax models for most datasets. Optimal values of tuning parameters \( \epsilon \) and \( \gamma \) are also shown in Tables 2 and 3. It is also interesting to point out that SAT-RF-y and Softmax using the RF provides the same measures \( R^2 \) or MAE for datasets Diabetes and Sparse. Indeed, it can be seen from Table 2 that \( \epsilon_{\text{opt}} = \gamma_{\text{opt}} = 0 \). This implies that only softmax operations without trainable parameters define the attention weights. It can be seen from Table 3 that this case does not take place for SAT-RF-y using ERT.

---

\(^2\)https://www.stat.berkeley.edu/$\sim$breiman/bagging.pdf.
The next modification for studying is SAT-RF-x. The corresponding results of numerical experiments under the same condition as experiments with SAT-RF-y are shown in Tables 4 and 5. However, if to compare these results with results given in Tables 2 and 3, then they are mainly inferior to SAT-RF-y and comparable to this modification when the RF is used. The same can be seen from Table 8 where SAT-RF-x is compared with SAT-RF-y and SAT-RF-yx. The results can be explained as follows. A large distance between $A_{i}(x)$ and $A_{j}(x)$ mainly says about a large difference between subsets of examples used for training the $i$-th and the $j$-th trees. However, this distance does not say about predictions of trees which are transformed by using the self-attention.

Results of numerical experiments with SAT-RF-yx are presented in Tables 6 and 7. One can see from the tables that SAT-RF-yx outperforms other models. In particular, it is shown in Table 6 that SAT-RF-yx provides better results for all datasets except for the Wine dataset. The same can be said about models constructed by using ERTs. The corresponding results are shown in Table 7. If we compare results from Table 6 with results from Table 7, then it is interesting to point out that the use of ERTs significantly improves the models.

Figure 1 illustrates how measure $R^2$ depends on the attention parameters $\tau$, $\epsilon$ (the left picture) and on the self-attention parameters $\kappa$, $\gamma$ (the right picture) for the Sparse dataset. It is interesting to see from Fig. 1 that $R^2$ achieves its maximum by $\tau = 1$ and $\epsilon = 0$ or $\epsilon = 0.25$. At the same time, $R^2$ achieves its maximum by $\kappa < 1$ and $\gamma = 0.25$. The optimal values $\epsilon$ and $\gamma$ coincide with the corresponding optimal values shown in Table 7. Figures 2, 3 and 4 illustrate the same dependencies for the Friedman 1, Boston, Wine datasets, respectively.

The best results for all modifications are illustrated in Table 8 which aims to compare different modifications of SAT-RF. We can see that SAT-RF-yx outperforms other models for most dataset when RFs are used. The same cannot be concluded when ERTs are used. Indeed, we can see from Table 8 that SAT-RF-yx based on ERTs is inferior other model in half of cases.

---

### Table 2
Measures $R^2$ and MAE for comparison of models (the RF, the Softmax model, SAT-RF-y) trained on regression datasets

| Dataset     | $\epsilon_{opt}$ | $\gamma_{opt}$ | $R^2$       | MAE       |
|-------------|------------------|----------------|-------------|-----------|
|             |                  |                | RF | Softmax | SAT-RF-y | RF | Softmax | SAT-RF-y |
| Diabetes    | 0                | 0              | 0.416 | 0.424 | 0.424 | 44.92 | 44.66 | 44.66 |
| Friedman 1  | 0.5              | 0              | 0.841 | 0.849 | 0.878 | 111.7 | 109.5 | 100.9 |
| Friedman 2  | 0.5              | 0              | 0.625 | 0.625 | 0.682 | 0.155 | 0.156 | 0.133 |
| Friedman 3  | 0.5              | 0.25           | 0.380 | 0.367 | 0.454 | 109.1 | 110.2 | 100.5 |
| Regression  | 0.5              | 0              | 0.814 | 0.818 | 0.823 | 2.539 | 2.508 | 2.494 |
| Sparse      | 0                | 0              | 0.470 | 0.522 | 0.522 | 1.908 | 1.802 | 1.802 |
| Airfoil     | 1                | 0.75           | 0.823 | 0.820 | 0.843 | 2.203 | 2.231 | 2.069 |
| Boston      | 0                | 0.5            | 0.845 | 0.841 | 0.857 | 4.855 | 4.948 | 4.694 |
| Concrete    | 0                | 1              | 0.981 | 0.981 | 0.989 | 1.004 | 1.006 | 0.787 |
| Wine        | 0.25             | 0.25           | 0.433 | 0.423 | 0.424 | 0.451 | 0.460 | 0.459 |
| Yacht       | 1                | 0.5            | 0.981 | 0.981 | 0.989 | 1.004 | 1.006 | 0.787 |

### Table 3
Measures $R^2$ and MAE for comparison of models (the ERT, the Softmax model, SAT-RF-y) trained on regression datasets

| Dataset     | $\epsilon_{opt}$ | $\gamma_{opt}$ | $R^2$       | MAE       |
|-------------|------------------|----------------|-------------|-----------|
|             |                  |                | ERT | Softmax | SAT-RF-y | ERT | Softmax | SAT-RF-y |
| Diabetes    | 0.75             | 0              | 0.456 | 0.458 | 0.453 | 44.50 | 44.38 | 44.51 |
| Friedman 1  | 1                | 0.75           | 0.471 | 0.471 | 0.521 | 2.502 | 2.502 | 2.414 |
| Friedman 2  | 1                | 0.25           | 0.813 | 0.813 | 0.939 | 123.03 | 122.66 | 73.77 |
| Friedman 3  | 0                | 1              | 0.570 | 0.570 | 0.739 | 0.179 | 0.179 | 0.138 |
| Regression  | 1                | 0              | 0.402 | 0.403 | 0.455 | 106.3 | 106.2 | 101.8 |
| Sparse      | 0                | 0.25           | 0.452 | 0.514 | 0.531 | 1.994 | 1.870 | 1.830 |
| Airfoil     | 1                | 0.75           | 0.802 | 0.802 | 0.837 | 2.370 | 2.370 | 2.127 |
| Boston      | 0.5              | 0.75           | 0.831 | 0.833 | 0.837 | 2.481 | 2.467 | 2.453 |
| Concrete    | 0                | 1              | 0.851 | 0.851 | 0.869 | 4.892 | 4.892 | 4.650 |
| Wine        | 0                | 0.25           | 0.418 | 0.418 | 0.419 | 0.464 | 0.463 | 0.463 |
| Yacht       | 0                | 1              | 0.988 | 0.988 | 0.988 | 0.824 | 0.824 | 0.818 |
Table 4 Measures $R^2$ and MAE for comparison of models (the RF, the Softmax model, SAT-RF-x) trained on regression datasets

| Dataset    | $\epsilon_{opt}$ | $\gamma_{opt}$ | $R^2$ RF | Softmax | SAT-RF-x | MAE RF | Softmax | SAT-RF-x |
|------------|-----------------|----------------|---------|---------|---------|--------|---------|---------|
| Diabetes   | 0               | 0              | 0.405   | 0.416   | 0.416  | 44.92  | 44.87   | 44.87   |
| Friedman 1 | 0               | 1              | 0.459   | 0.438   | 0.470  | 2.540  | 2.589   | 2.540   |
| Friedman 2 | 1               | 1              | 0.841   | 0.834   | 0.872  | 111.7  | 114.5   | 103.7   |
| Friedman 3 | 0.5             | 0.5            | 0.625   | 0.623   | 0.684  | 0.154  | 0.156   | 0.134   |
| Regression | 0.75            | 0              | 0.380   | 0.374   | 0.451  | 109.1  | 110.0   | 100.4   |
| Sparse     | 0               | 0              | 0.470   | 0.488   | 0.488  | 1.908  | 1.860   | 1.860   |
| Airfoil    | 0.25            | 1              | 0.823   | 0.820   | 0.843  | 2.203  | 2.231   | 2.070   |
| Boston     | 0.25            | 0.5            | 0.814   | 0.814   | 0.821  | 2.539  | 2.539   | 2.518   |
| Concrete   | 0               | 1              | 0.845   | 0.841   | 0.857  | 4.855  | 4.948   | 4.694   |
| Wine       | 0               | 0.75           | 0.433   | 0.421   | 0.422  | 0.451  | 0.461   | 0.459   |
| Yacht      | 0               | 1              | 0.981   | 0.981   | 0.989  | 1.004  | 1.004   | 0.787   |

Table 5 Measures $R^2$ and MAE for comparison of models (the ERT, the Softmax model, SAT-RF-x) trained on regression datasets

| Dataset    | $\epsilon_{opt}$ | $\gamma_{opt}$ | $R^2$ ERT | Softmax | SAT-RF-x | MAE ERT | Softmax | SAT-RF-x |
|------------|-----------------|----------------|---------|---------|---------|--------|---------|---------|
| Diabetes   | 0               | 0.75           | 0.449   | 0.449   | 0.442  | 44.41  | 44.37   | 44.57   |
| Friedman 1 | 0               | 1              | 0.471   | 0.471   | 0.513  | 2.502  | 2.50    | 2.426   |
| Friedman 2 | 0               | 1              | 0.813   | 0.813   | 0.930  | 123.0  | 123.0   | 74.50   |
| Friedman 3 | 0               | 1              | 0.570   | 0.570   | 0.739  | 0.179  | 0.179   | 0.138   |
| Regression | 1               | 0              | 0.402   | 0.443   | 0.493  | 106.3  | 102.5   | 95.95   |
| Sparse     | 0               | 0.25           | 0.452   | 0.501   | 0.518  | 1.994  | 1.887   | 1.851   |
| Airfoil    | 0.5             | 1              | 0.802   | 0.802   | 0.837  | 2.370  | 2.370   | 2.128   |
| Boston     | 1               | 0.25           | 0.831   | 0.835   | 0.843  | 2.481  | 2.447   | 2.402   |
| Concrete   | 0               | 1              | 0.851   | 0.851   | 0.863  | 4.892  | 4.892   | 4.650   |
| Wine       | 1               | 0              | 0.418   | 0.417   | 0.418  | 0.462  | 0.463   | 0.462   |
| Yacht      | 0               | 1              | 0.988   | 0.988   | 0.988  | 0.824  | 0.824   | 0.818   |

Table 6 Measures $R^2$ and MAE for comparison of models (the RF, the Softmax model, SAT-RF-xy) trained on regression datasets

| Dataset    | $\epsilon_{opt}$ | $\gamma_{opt}$ | $R^2$ RF | Softmax | SAT-RF-xy | MAE RF | Softmax | SAT-RF-xy |
|------------|-----------------|----------------|---------|---------|-----------|--------|---------|-----------|
| Diabetes   | 0               | 0              | 0.416   | 0.422   | 0.422    | 44.92  | 45.01   | 45.01     |
| Friedman 1 | 1               | 0.75           | 0.459   | 0.440   | 0.489    | 2.540  | 2.574   | 2.509     |
| Friedman 2 | 1               | 0.5            | 0.841   | 0.788   | 0.882    | 111.7  | 125.0   | 95.84     |
| Friedman 3 | 0.5             | 0.5            | 0.625   | 0.628   | 0.685    | 0.154  | 0.155   | 0.133     |
| Regression | 1               | 0.25           | 0.380   | 0.363   | 0.488    | 109.1  | 111.4   | 96.50     |
| Sparse     | 0               | 0              | 0.470   | 0.531   | 0.540    | 1.908  | 1.783   | 1.775     |
| Airfoil    | 0.25            | 1              | 0.823   | 0.820   | 0.849    | 2.203  | 2.231   | 2.070     |
| Boston     | 0.25            | 0.75           | 0.814   | 0.814   | 0.824    | 2.539  | 2.546   | 2.501     |
| Concrete   | 1               | 0.75           | 0.845   | 0.841   | 0.866    | 4.834  | 4.921   | 4.651     |
| Wine       | 0.25            | 0.5            | 0.433   | 0.422   | 0.429    | 0.451  | 0.461   | 0.458     |
| Yacht      | 1               | 0.5            | 0.981   | 0.971   | 0.989    | 1.004  | 1.237   | 0.790     |
### Table 7

Measures $R^2$ and MAE for comparison of models (the ERT, the Softmax model, SAT-RF-yx) trained on regression datasets

| Dataset   | $\epsilon_{opt}$ | $\gamma_{opt}$ | ERT | Softmax | SAT-RF-yx | ERT | Softmax | SAT-RF-yx |
|-----------|------------------|----------------|-----|---------|-----------|-----|---------|-----------|
| Diabetes  | 0                | 0              | 0.438 | 0.439 | 0.439    | 44.55 | 44.26   | 44.26     |
| Friedman 1| 0                | 1              | 0.471 | 0.471 | 0.513    | 2.502 | 2.502   | 2.426     |
| Friedman 2| 1                | 0              | 0.813 | 0.813 | 0.930    | 123.0 | 123.0   | 74.49     |
| Friedman 3| 1                | 0.5            | 0.570 | 0.570 | 0.751    | 0.179 | 0.179   | 0.137     |
| Regression| 0                | 0.75           | 0.402 | 0.411 | 0.449    | 106.3 | 105.4   | 101.1     |
| Sparse    | 0                | 0.25           | 0.452 | 0.518 | 0.542    | 1.994 | 1.863   | 1.822     |
| Airfoil   | 0.5              | 1              | 0.802 | 0.802 | 0.841    | 2.370 | 2.370   | 2.128     |
| Boston    | 0.75             | 0              | 0.831 | 0.836 | 0.844    | 2.481 | 2.452   | 2.427     |
| Concrete  | 1                | 0.5            | 0.839 | 0.839 | 0.860    | 5.119 | 5.128   | 4.689     |
| Wine      | 0                | 0.75           | 0.418 | 0.417 | 0.447    | 0.464 | 0.463   | 0.462     |
| Yacht     | 0                | 1              |        | 0.988 | 0.988    | 0.824 | 0.824   | 0.818     |

### Table 8

Measure $R^2$ for comparison of models (SAT-RF-y, SAT-RF-x, SAT-RF-yx) trained on regression datasets

| Dataset   | RF    | SAT-RF-y | SAT-RF-x | SAT-RF-yx | ERT    | SAT-RF-y | SAT-RF-x | SAT-RF-yx |
|-----------|-------|----------|----------|-----------|--------|----------|----------|-----------|
| Diabetes  | 0.424 | 0.416    | 0.422    | 0.453     | 0.442  | 0.439    |          |           |
| Friedman  1| 0.470 | 0.470    | 0.489    | 0.521     | 0.513  | 0.513    |          |           |
| Friedman  2| 0.878 | 0.872    | 0.882    | 0.939     | 0.930  | 0.930    |          |           |
| Friedman  3| 0.682 | 0.684    | 0.685    | 0.739     | 0.739  | 0.751    |          |           |
| Regression| 0.454 | 0.483    | 0.488    | 0.455     | 0.493  | 0.449    |          |           |
| Sparse    | 0.522 | 0.488    | 0.540    | 0.531     | 0.518  | 0.542    |          |           |
| Airfoil   | 0.823 | 0.820    | 0.849    | 0.802     | 0.802  | 0.841    |          |           |
| Boston    | 0.823 | 0.821    | 0.824    | 0.837     | 0.843  | 0.844    |          |           |
| Concrete  | 0.857 | 0.857    | 0.866    | 0.869     | 0.863  | 0.860    |          |           |
| Wine      | 0.424 | 0.422    | 0.429    | 0.419     | 0.416  | 0.447    |          |           |
| Yacht     | 0.989 | 0.989    | 0.989    | 0.988     | 0.988  | 0.988    |          |           |

### Fig. 1

Measure $R^2$ as a function of the attention parameters $\tau$, $\epsilon$ (left) and the self-attention parameters $\kappa$, $\gamma$ (right) for the Sparse dataset and SAT-RF-yx using the ERT.
It should be noted that SAT-RF is an extension of ABRF under condition that the self-attention is supplemented. Therefore, we compare results of ABRF and SAT-RF for two cases when RFs and ERTs are used. Moreover, we compare SAT-RF with ϵ-ABRF because SAT-RF is based on this modification of ABRF and the attention-based models ϵ-ABRF and Softmax because these models are studied in [55] in detail. Moreover, it is illustrated in [55] that ϵ-ABRF outperforms RF as well as ERT for all considered datasets. Therefore, it is interesting to find out whether modifications of SAT-RF provide better results in comparison with ϵ-ABRF. It is also important to point out that we did not consider a more complex attention-based model proposed in [55] called ϵ-σ-ABRF, which uses the gradient-based algorithm to learn the attention parameters. However, if we compare results provided by this model and demonstrated in [55], then we can see that results of SAT-RF are again better in comparison with ϵ-σ-ABRF. The outperformance of SAT-RF can be explained by its robustness to possible perturbations of vectors $x$ and values $y$ or to outliers of in data and predictions provided by each decision tree. The robustness is achieved by applying the self-attention mechanism which aims to remove the noise or outliers in data and the tree predictions.

The best results of both models (SAT-RF and ϵ-ABRF) are shown in Table 9. It can be seen from Table 9 that adding the self-attention module to ABRF improves the results. To formally test whether the average difference in the performance of two models, SAT-RF and ϵ-ABRF, is significantly differ-
Measure $R^2$ as a function of the attention parameters $\tau$, $\epsilon$ (left) and the self-attention parameters $\kappa$, $\gamma$ (right) for the Wine dataset and SAT-RF-yy using the ERT

Table 9: Measure $R^2$ for comparison of models SAT-RF and $\epsilon$-ABRF trained on regression datasets

| Dataset     | RF   | $\epsilon$-ABRF | SAT-RF | ERT   | $\epsilon$-ABRF | SAT-RF |
|-------------|------|------------------|--------|-------|------------------|--------|
| Diabetes    | 0.424| 0.424            | 0.441  | 0.453 |
| Friedman 1  | 0.470| 0.489            | 0.513  | 0.521 |
| Friedman 2  | 0.877| 0.882            | 0.930  | 0.939 |
| Friedman 3  | 0.686| 0.685            | 0.739  | 0.751 |
| Regression  | 0.450| 0.488            | 0.447  | 0.493 |
| Sparse      | 0.529| 0.540            | 0.536  | 0.542 |
| Airfoil     | 0.843| 0.849            | 0.837  | 0.841 |
| Boston      | 0.823| 0.824            | 0.838  | 0.844 |
| Concrete    | 0.857| 0.866            | 0.863  | 0.869 |
| Wine        | 0.423| 0.429            | 0.416  | 0.447 |
| Yacht       | 0.989| 0.989            | 0.988  | 0.988 |

8 Concluding remarks

Extensions of the attention-based RF based on joint use of the attention and self-attention mechanisms have been proposed. The attention part plays role of assigning weights to decision trees in the RF, and the self-attention part tries to capture dependencies of the tree predictions and to remove noise or anomalous predictions. They can be regarded as an alternative tool for handling tabular data. The proposed models allow us to avoid using neural networks and gradient-based algorithms. One of the important peculiarities of the models is that the attention and the self-attention are learned jointly by solving the quadratic problem with the attention and self-attention weights as optimization variables.

Advantages of the proposed models are the following. First, the models are simply learned. Second, in contrast to neural networks, the models have a few hyperparameters: parameters of the Gaussian kernels (softmax operations) and the contamination parameters of the Huber’s $\epsilon$-contamination model. Third, the attention part allows us to improve predictions and the self-attention part allows to take into account “bad” trees and anomalous predictions. Results of numerical experiments clearly illustrate the above. The proposed models can be extended by adding new self-attention modules which form the multi-head self-attention. However, this extension is rather complex from the implementation point of view.

We have to point out also disadvantages. First, the proposed models are mainly restricted by dealing with tabular data due to the RF basis of the models. Second, in contrast to the attention-based RF, the model has a larger number of training parameters (weights of the attention and the self-
attention). If the number of trees in the RF is rather large, then the number of training parameters significantly increases. This may lead to overfitting. Third, the advantage of the models to handle tabular data can be viewed as its disadvantage because other types of data, for instance, images, graphs, text data may result worse predictions.

Many numerical experiments have demonstrated the outperformance of the proposed models. Moreover, the results have demonstrated that SAT-RFs outperform the attention-based RFs which are the basis for the proposed models. At the same time, we have considered only datasets with rather small numbers of features. A reason for studying these datasets is that the attention and self-attention mechanisms are applied to RFs which aim to deal mainly with small tabular data. Nevertheless, we have to point out that the proposed ideas can be inherently extended to the so-called deep forest [76] which is an ensemble of RFs and can be successfully used for dealing with high-dimensional data, for example, with images. The incorporation of the attention and self-attention mechanisms into the deep forest is an interesting direction for further research.

Due to flexibility of SAT-RFs, many modifications can be proposed and studied, for example, various kernel functions, models of weights different from the Huber’s $\epsilon$-contamination model. Attention weights as well as self-attention weights can be assigned to subsets of trees. This approach allows us to partially reduce the number of training parameters. It is interesting to develop algorithms for implementing the multi-head self-attention. All the above ideas can be also regarded as directions for further research.

Acknowledgements The authors would like to express their appreciation to the anonymous referees whose very valuable comments have improved the paper.

Funding This work is supported by the Russian Science Foundation under grant 21-11-00116.

Data availability Data are available from open sources.

Declarations

Conflict of interest I certify that no party having a direct interest in the results of the research supporting this article has or will confer a benefit on me or on any organization with which I am associated, and I certify that all financial and material supports for this research and work are clearly identified in the title page of the manuscript.

Code availability The corresponding code implementing the method is publicly available https://github.com/andruekonst/forest-self-attention.

References

1. Arik, S., Pfister, T.: Tabnet: Attentive interpretable tabular learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 6679–6687 (2021)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. Preprint at arXiv:1409.0473 (2014)
3. Beltagy, I., Peters, M., Cohan, A.: Longformer: The long-document transformer. Preprint at arXiv:2004.05150 (2020)
4. Borisov, V., Leemann, T., Sessler, K., et al.: Deep neural networks and tabular data: A survey. Preprint at arXiv:2110.01889v2 (2021)
5. Brauwers, G., Frasincar, F.: A general survey on attention mechanisms in deep learning. Preprint at arXiv:2203.14263 (2022)
6. Breiman, L.: Random forests. Mach. Learn. 45(1), 5–32 (2001)
7. Chaudhari, S., Mithal, V., Polatkan, G., et al.: An attentive survey of attention models. Preprint at arXiv:1904.02874 (2019)
8. Chen, Z., Xie, L., Niu, J., et al.: Joint self-attention and scale-aggregation for self-calibrated deraining network. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 2517–2525 (2020)
9. Cheng, J., Dong, L., Lapata, M.: Long short-term memory-networks for machine reading. Preprint at arXiv:1601.06733 (2016)
10. Choromanski, K., Chen, H., Lin, H., et al.: Hybrid random features. Preprint at arXiv:2110.04367v2 (2021a)
11. Choromanski, K., Likhosherstov, V., Dohan, D., et al.: Rethinking attention with performers. In: 2021 International Conference on Learning Representations, pp. 1–38 (2021b)
12. Correia, A., Colombini, E.: Attention, please! A survey of neural attention models in deep learning. Preprint at arXiv:2103.16775 (2021a)
13. Correia, A., Colombini, E.: Neural attention models in deep learning: survey and taxonomy. Preprint at arXiv:2112.05909 (2021b)
14. Daho, M., Settouti, N., Lazouni, M., et al.: Weighted vote for trees aggregation in random forest. In: 2014 International Conference on Multimedia Computing and Systems (ICMCS). IEEE, pp. 438–443 (2014)
15. Dai, Z., Yang, Z., Yang, Y., et al.: Transformer-xl: attentive language models beyond a fixed-length context. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 2978–2988 (2019)
16. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7, 1–30 (2006)
17. Devlin, J., Chang, M., Lee, K., et al.: Bert: Pre-training of deep bidirectional transformers for language understanding. Preprint at arXiv:1810.04805 (2018)
18. Dua, D., Graff, C.: UCI machine learning repository. (2017). http://archive.ics.uci.edu/ml
19. Fournier, Q., Caron, G., Aloise, D.: A practical survey on faster and lighter transformers. Preprint at arXiv:2103.14636 (2021)
20. Friedman, J.: Greedy function approximation: a gradient boosting machine. Ann. Stat. 29, 1189–1232 (2001)
21. Friedman, J.: Stochastic gradient boosting. Comput. Stat. Data Anal. 38(4), 367–378 (2002)
22. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. 63, 3–42 (2006)
23. Goncalves, T., Rio-Torto, I., Teixeira, L., et al.: A survey on attention mechanisms for medical applications: are we moving towards better algorithms?. Preprint at arXiv:2204.12406 (2022)
24. Guo, MH., Liu, ZN., Mu, T.J., et al.: Beyond self-attention: external attention using two linear layers for visual tasks. Preprint at arXiv:2105.02358 (2021)

© Springer
25. Hassanin, M., Anwar, S., Radwan, I., et al.: Visual attention methods in deep learning: an in-depth survey. Preprint at arXiv:2204.07756 (2022)
26. Huber, P.: Robust Statistics. Wiley, New York (1981)
27. Katzir, L., Eldian, G., El-Yaniv, R.: Net-dnf: effective deep modeling of tabular data. In: 9th International Conference on Learning Representations, ICLR 2021, pp. 1–16 (2021)
28. Khan, S., Naseer, M., Hayat, M., et al.: Transformers in vision: a survey. ACM Comput. Surv. 54, 1–41 (2022)
29. Kim, H., Kim, H., Moon, H., et al.: A weight-adjusted voting algorithm for ensemble of classifiers. J. Korean Stat. Soc. 40(4), 437–449 (2011)
30. Konstantinov, A., Utkin, L., Kirpichenko, S.: AGBoost: attention-based modification of gradient boosting machine. In: 31st Conference of Open Innovations Association (FRUCT). IEEE, pp. 96–101 (2022)
31. Li, H.B., Wang, W., Ding, H.W., et al.: Trees weighting random forest method for classifying high-dimensional noisy data. In: 2010 IEEE 7th International Conference on E-Business Engineering. IEEE, pp. 160–163 (2010)
32. Li, M., Hsu, W., Xie, X., et al.: SACNN: Self-attention convolutional neural network for low-dose CT denoising with self-supervised perceptual loss network. IEEE Trans. Med. Imaging 39(7), 2289–2301 (2020)
33. Lin, T., Wang, Y., Liu, X., et al.: A survey of transformers. Preprint at arXiv:2106.04554 (2021)
34. Lin, Z., Feng, M., dos Santos, C., et al.: A structured self-attentive sentence embedding. In: The 5th International Conference on Learning Representations (ICLR 2017), pp. 1–15 (2017)
35. Liu, F., Huang, X., Chen, Y., et al.: Random features for kernel approximation: A survey on algorithms, theory, and beyond. Preprint at arXiv:2004.11154v5 (2021a)
36. Liu, Z., Lin, Y., Cao, Y., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). IEEE, pp. 10,012–10,022 (2021b)
37. Luong, T., Pham, H., Manning, C.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. The Association for Computational Linguistics, pp. 1412–1421 (2015)
38. Ma, X., Kong, X., Wang, S., et al.: Luna: Linear unified nested attention. Preprint at arXiv:2106.01540 (2021)
39. Nadaraya, E.: On estimating regression. Theory Probab. Appl. 9(1), 141–142 (1964)
40. Niu, Z., Zhong, G., Yu, H.: A review on the attention mechanism of deep learning. Neurocomputing 452, 48–62 (2021)
41. Parikh, A., Tackstrom, O., Das, D., et al.: A decomposable attention model for natural language inference. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp. 2249–2255 (2016)
42. Parmar, N., Vaswani, A., Uszkoreit, J., et al.: Image transformer. In: International Conference on Machine Learning. PMLR, pp. 4055–4064 (2018)
43. Peng, H., Pappas, N., Yogatama, D., et al.: Random feature attention. In: International Conference on Learning Representations (ICLR 2021), pp. 1–19 (2021)
44. Povey, D., Hadian, H., Ghahremani, P., et al.: A time-restricted self-attention layer for ASR. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 5874–5878 (2018)
45. Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A., Shlens, J.: Stand-alone self-attention in vision models. In: Proceedings of the 33rd Conference on Neural Information Processing Systems, pp. 1–13 (2019)
46. Ronao, C., Cho, S.B.: Random forests with weighted voting for anomalous query access detection in relational databases. In: Artificial Intelligence and Soft Computing. ICAISC 2015, Lecture Notes in Computer Science, vol. 9120, pp. 36–48. Springer, Cham (2015)
47. Schlag, I., Irie, K., Schmidhuber, J.: Linear transformers are secretly fast weight programmers. In: International Conference on Machine Learning 2021. PMLR, pp. 9355–9366 (2021)
48. Shen, Z., Bello, I., Vemulapalli, R., et al.: Global self-attention networks for image recognition. Preprint at arXiv:2010.03019 (2020)
49. Shim, K., Choi, J., Sung, W.: Understanding the role of self attention for efficient speech recognition. In: The Tenth International Conference on Learning Representations (ICLR), pp. 1–19 (2022)
50. Shwartz-Ziv, R., Amitai, A.: Tabular data: deep learning is not all you need. Inf. Fus. 81, 84–90 (2022)
51. Somepalli, G., Goldblum, M., Schwarzchild, A., et al.: Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. Preprint at arXiv:2106.01342 (2021)
52. Soysadan, D.: Attention mechanism in neural networks: where it comes and where it goes. Preprint at arXiv:2204.13154 (2022)
53. Tay, Y., Dehghani, M., Bahri, D., et al.: Efficient transformers: a survey. ACM Comput. Surv. 55(6), 1–28 (2022)
54. Tian, C., Fei, L., Zheng, W., et al.: Deep learning on image denoising: an overview. Neural Netw. 131, 251–275 (2020)
55. Utkin, L., Konstantinov, A.: Attention-based random forest and contamination model. Neural Netw. 154, 346–359 (2022)
56. Utkin, L., Konstantinov, A., Chukanov, V., et al.: A weighted random survival forest. Knowl.-Based Syst. 177, 136–144 (2019)
57. Utkin, L., Kovalov, M., Meldo, A.: A deep forest classifier with weights of class probability distribution subsets. Knowl.-Based Syst. 173, 15–27 (2019)
58. Utkin, L., Konstantinov, A., Chukanov, V., et al.: A new adaptive weighted deep forest and its modifications. Int. J. Inform. Technol. Decis. Mak. 19(4), 963–986 (2020)
59. Utkin, L., Kovalov, M., Coolen, F.: Imprecise weighted extensions of random forests for classification and regression. Appl. Soft Comput. 92(106324), 1–14 (2020)
60. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30, Curran Associates pp. 5998–6008, (2017)
61. Vidal, R.: Attention: Self-expression is all you need, iCLR 2022, OpenReview.net. https://openreview.net/forum?id=MmujBClawFo (2022)
62. Vyas, A., Katharopoulos, A., Fleuret, F.: Fast transformers with clustered attention. In: Advances in Neural Information Processing Systems 33, pp. 21665–21674 (2020)
63. Wang, F., Jiang, M., Qian, C., et al.: Residual attention network for image classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3156–3164 (2017)
64. Wang, X., Girshick, R., Gupta, A., et al.: Non-local neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7794–7803 (2018)
65. Watson, G.: Smooth regression analysis. Sankhya: Indian J. Stat. Ser. A 26, 359–372 (1964)
66. Winham, S., Freimuth, S.: Refined weighted random forest and its application to credit card fraud detection. In: Computational Data and Social Networks, pp. 343–355. Springer International Publishing, Cham (2018)
70. Yu, J., Nie, Y., Long, C., et al.: Monte Carlo denoising via auxiliary feature guided self-attention. ACM Trans. Gr. 40(6), 1–13 (2021)
71. Zhang, A., Lipton, Z., Li, M., et al.: Dive into deep learning. Preprint at arXiv:2106.11342 (2021)
72. Zhang, H., Quost, B., Masson, M.H.: Cautious weighted random forests. Expert Syst. Appl. 213, 118883 (2023)
73. Zhang, X., Wang, M.: Weighted random forest algorithm based on bayesian algorithm. In: Journal of Physics: Conference Series, vol 1924. IOP Publishing, p. 012006 (2021)
74. Zhao, H., Jia, J., Koltun, V.: Exploring self-attention for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10076–10085 (2020)
75. Zheng, L., Wang, C., Kong, L.: Linear complexity randomized self-attention mechanism. In: Proceedings of the 39th International Conference on Machine Learning. PMLR, pp. 27011–27041 (2022)
76. 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 (IJCAI’17). AAAI Press, Melbourne, Australia, pp. 3553–3559 (2017)
77. Zuo, Z., Chen, X., Xu, H., et al.: Idea-net: Adaptive dual self-attention network for single image denoising. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops, pp. 739–748 (2022)

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

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