AGBoost: Attention-based Modification of Gradient Boosting Machine

Andrei Konstantinov, Lev Utkin, Stanislav Kirpichenko
Peter the Great Saint-Petersburg Polytechnic University,
Saint-Petersburg, Russia
andrue.konst@gmail.com, lev.utkin@gmail.com, kirpichenko.sr@gmail.com

***Abstract*—A new attention-based model for the gradient boosting machine (GBM) called AGBoost (the attention-based gradient boosting) is proposed for solving regression problems. The main idea behind the proposed AGBoost model is to assign attention weights with trainable parameters to iterations of GBM under condition that decision trees are base learners in GBM. Attention weights are determined by applying properties of decision trees and by using the Huber's contamination model which provides an interesting linear dependence between trainable parameters of the attention and the attention weights. This peculiarity allows us to train the attention weights by solving the standard quadratic optimization problem with linear constraints. The attention weights also depend on the discount factor as a tuning parameter, which determines how much the impact of the weight is decreased with the number of iterations. Numerical experiments performed for two types of base learners, original decision trees and extremely randomized trees with various regression datasets illustrate the proposed model.**

I. Introduction

One of the promising tools in deep learning is the attention mechanism which assigns weight to instances or features in accordance with their importance for enhancing the regression and classification performance. The attention mechanism comes from the biological nature of the human perception to be concentrated on some important parts of images, text, data, etc. [1]. Following this property, various models of attention have been developed in order to improve machine learning models. Many interesting surveys devoted to different forms of the attention mechanism, including transformers as the powerful neural network models, can be found in [1]–[5].

An important peculiarity of the attention mechanism is that it is trainable, i.e., it, as a model, contains trainable parameters. Due to this property most attention models are components of neural networks [2], and the attention trainable parameters are learned by using the gradient-based algorithms which may lead to overfitting, expensive computations, i.e., the attention models have the same problems as neural networks. In order to overcome this difficulty and simultaneously to get attention-based models with a simple training algorithm, Utkin and Konstantinov [6] proposed a new model which is called the attention-based random forest. According to this model, the attention weights are assigned to decision trees in the random forest [7] in a specific way. Moreover, the attention weights have trainable parameters which are learned on the corresponding dataset. One of the main ideas behind the attention-based random forest is to apply the Huber's ϵ-contamination model [8] which establishes relationship between the attention weights and trainable parameters. Various numerical examples with well-known regression and classification datasets demonstrated outperforming results.

The random forest is a powerful ensemble-based model which is especially efficient when we deal with tabular data. However, there is another ensemble-based model, the well-known gradient boosting machine (GBM) [9], [10], which is reputed a more efficient model for many datasets and more flexible one. GBMs have illustrated their efficiency for solving regression problems [11].

Following the attention-based random forest model, we aim to apply some ideas behind this model to the GBM and to develop quite a new model called the attention-based gradient boosting machine (AGBoost). In accordance with AGBoost, we assign weights to each iteration of the GBM in a specific way taking into account the tree predictions and the discount factor which determines how much the impact of the attention weight is decreased with the number of iterations. It is important to note that the attention mechanism [12] was originally represented in the form of the Nadaraya-Watson kernel regression model [13], [14], where attention weights conform with relevance of a training instance to a target feature vector. The idea behind AGBoost is to incorporate the Nadaraya-Watson kernel regression model into the GBM. We also apply the Huber's ϵ-contamination model where the contamination distribution over all iterations is regarded as a trainable parameter vector. The training process of attention weights is reduced to solving the standard quadratic optimization problem with linear constraints. We consider AGBoost only for solving regression problems. However, the results can be simply extended to classification problems.

Numerical experiments with regression datasets are provided for studying the proposed attention-based model. Two types of decision trees are used in experiments: original decision trees and Extremely Randomized Trees (ERT) proposed in [15]. At each node, the ERT algorithm chooses a split point randomly for each feature and then selects the best split among these.

The paper is organized as follows. Related work can be found in Section 2. A brief introduction to the attention mechanism is given in Section 3. The proposed AGBoost model for regression is presented in Section 4. Numerical experiments illustrating regression problems are provided in Section 5. Concluding remarks can be found in Section 6.

ISSN 2305-7254

II. Related work

Attention mechanism. The attention mechanism is considered a powerful and perspective tool for constructing machine learning models having accurate performance in several applications. As a result, many classification and regression algorithms have been added by attention-based models to improve their performance. Attention models became the key modules of Transformers [5], which have achieved great success in many applications and fields, including natural language processing and computer vision. Surveys of various attention-based models can be found in [1]–[4], [16]. In spite of efficiency of the attention models, they require to train the softmax functions with trainable parameters that leads to computational problems. Several methods of the softmax function linearization were developed [17]–[20].

Our aim is to propose the attention-based GBM modification which train the attention parameters by means of the quadratic optimization that is simply solved.

Gradient boosting machines. The GBM is one of the most efficient tool for solving regression and classification problems especially with tabular data. Moreover, it can cope with non-linear dependencies [21]. Decision trees are often used in GBMs as basic models. The original GBM [9] is based on decision trees which are sequentially trained to approximate negative gradients. Due to the success of GBMs, various modifications have been developed, for example, the well-known XGBoost [22], pGBRT [23], SGB [10]. Advantages of decision trees in GBMs led to a modification of the GBM on the basis of the deep forests [24], which is called the multi-layered gradient boosting decision tree model [25]. An interesting modification is the soft GBM [26]. Another direction for modifying GBMs is to use extremely randomized trees [15] which illustrated the substantial improvement of the GBM performance. Several GBM models have been implemented by using modifications of extremely randomized trees and their modifications [27].

We modify the GBM to incorporate the attention mechanism into the iteration process and to weigh each iteration with respect to its importance in the final prediction.

III. Preliminary

A. The attention mechanism

The attention mechanism can be viewed as a trainable mask which emphasizes relevant information in a feature map. One of the clear explanations of the attention mechanism is to consider it from the statistics point of view [2], [12] in the form of the Nadaraya-Watson kernel regression model [13], [14].

Let us consider a dataset $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_n, y_n)\}$ consisting of n instances, where $\mathbf{x}_i = (x_{i1}, ..., x_{im}) \in \mathbb{R}^m$ is a feature vector involving m features, $y_i \in \mathbb{R}$ represents the regression target variable. The regression task is to learn a function $f : \mathbb{R}^m \to \mathbb{R}$ on the dataset D such that the trained function f can predict the target value $\tilde{y}$ of a new observation $\mathbf{x}$.

According to the Nadaraya-Watson regression model [13], [14], the target value y for a new vector of features $\mathbf{x}$ can be computed by using the weighted average of the form:

$$\tilde{y} = f(\mathbf{x}) = \sum_{i=1}^{n} \alpha(\mathbf{x}, \mathbf{x}_i) y_i. \tag{1}$$

Here weight $\alpha(\mathbf{x}, \mathbf{x}_i)$ indicates how close the vector $\mathbf{x}_i$ from the dataset D to the vector $\mathbf{x}$ that is the closer the vector $\mathbf{x}_i$ to $\mathbf{x}$, the greater the weight $\alpha(\mathbf{x}, \mathbf{x}_i)$ assigned to y_i.

The Nadaraya-Watson kernel regression uses kernels K in order to express weights $\alpha(\mathbf{x}, \mathbf{x}_i)$, in particular, the weights can be computed as:

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

If to apply terms introduced for the attention mechanism in [28], then weights $\alpha(\mathbf{x}, \mathbf{x}_i)$ are called as the attention weights, the target values y_i are called values, vectors $\mathbf{x}$ and $\mathbf{x}_i$ are called query and keys, respectively. It should be noted that the original Nadaraya-Watson kernel regression is a non-parametric model, i.e., it is an example of the non-parametric attention pooling. However, the weights can be added by trainable parameters which results the parametric attention pooling. In particular, one of the well-known kernels in (2) is the Gaussian kernel which produces the softmax function of the Euclidean distance. The attention weights with trainable parameters may have the form [28]:

$$\alpha(\mathbf{x}, \mathbf{x}_i) = \text{softmax}\left(\mathbf{q}^{\mathrm{T}}\mathbf{k}_i\right) = \frac{\exp\left(\mathbf{q}^{\mathrm{T}}\mathbf{k}_i\right)}{\sum_{j=1}^{n} \exp\left(\mathbf{q}^{\mathrm{T}}\mathbf{k}_j\right)}, \tag{3}$$

where $\mathbf{q} = \mathbf{W}_q\mathbf{x}$, $\mathbf{k}_i = \mathbf{W}_k\mathbf{x}_i$, $\mathbf{W}_q$ and $\mathbf{W}_k$ are matrices of trainable parameters.

Many definitions of attention weights and the attention mechanisms can be presented, for example, the additive attention [28], multiplicative or dot-product attention [29], [30]. A new attention mechanism is proposed below, which is based on training the weighted GBMs and the Huber's ϵ-contamination model.

B. A brief introduction to the GBM for regression

If to return to the regression problem stated above, then we aim to construct a regression model or an approximation g of the function f that minimizes the expected risk or the expected loss function

$$\begin{aligned} L(g) &= \mathbb{E}_{(\mathbf{x},y)\sim P}\ L(y, g(\mathbf{x})) \\ &= \int_{\mathcal{X}\times\mathbb{R}} L(y, g(\mathbf{x}))\mathrm{d}P(\mathbf{x}, y), \end{aligned} \tag{4}$$

with respect to the function parameters. Here $P(\mathbf{x}, y)$ is a joint probability distribution of $\mathbf{x}$ and y; the loss function $L(\cdot, \cdot)$ may be represented, for example, as follows:

$$L(y, g(\mathbf{x})) = (y - g(\mathbf{x}))^2. \tag{5}$$

Among many machine learning methods, which solve the regression problem, for example, random forests [7] and the

support vector regression [31]), the GBM [10] is one of the most accurate methods.

Generally, GBMs iteratively improve the predictions of y from $\mathbf{x}$ with respect to the loss function L. It is carried out by starting from an approximation of g, for example, from some constant c, and then adding new weak or base learners that improve upon the previous ones M times. As a result, an additive ensemble model of size M is formed:

$$g_0(\mathbf{x}) = c, \quad g_i(\mathbf{x}) = g_{i-1}(\mathbf{x}) + \gamma_i h_i(\mathbf{x}), \ i = 1, ..., M. \quad (6)$$

where h_i is the i-th base model at the i-th iteration; γ_i is the coefficient or the weight of the i-th base model.

Many GBMs use decision trees as the most popular base learners. The GBM is represented in the form of Algorithm 1. It can be seen from Algorithm 1 that it minimizes the expected loss function L by computing the gradient iteratively. Each decision tree in the GBM is constructed at each iteration to fit the negative gradients. The function h_i can be defined by parameters θ_i, i.e., $h_i(\mathbf{x}) = h(\mathbf{x}, \theta_i)$. It is trained on a new dataset $\{(\mathbf{x}_j, q_j^{(i)})\}$, where $q_j^{(i)}$, $j = 1, ..., n$, are residuals defined as partial derivatives of the expected loss function at each point $\mathbf{x}_i$ (see (7)).

Algorithm 1 The original GBM algorithm

Require: Training set D; the number of iterations T
Ensure: Prediction $g(\mathbf{x})$ for an instance $\mathbf{x}$

1: Initialize the function $g_0(\mathbf{x}) = c$
2: **for** $t = 1$, $t \leq T$ **do**
3: Calculate the residual $q_i^{(t)}$ as the partial derivative of the expected loss function $L(y_i, g_t(\mathbf{x}_i))$ at each point of the training set:

$$q_i^{(t)} = -\left.\frac{\partial L(y_i, z)}{\partial z}\right|_{z=g_{i-1}(\mathbf{x}_i)}, \ i = 1, ..., n \quad (7)$$

4: Train a base model $h_t(\mathbf{x}_i)$ on a new dataset with residuals $\{(\mathbf{x}_i, q_i^{(t)})\}$
5: Find the best gradient descent step-size γ_t:

$$\gamma_t = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_t, g_{t-1}(\mathbf{x}_i) + \gamma h_t(\mathbf{x}_i)) \quad (8)$$

6: Update the function $g_t(\mathbf{x}) = g_{t-1}(\mathbf{x}) + \gamma_t h_t(\mathbf{x})$
7: **end for**
8: The resulting function after T iterations is

$$g_T(\mathbf{x}) = \sum_{t=1}^{T} \gamma_t h_t(\mathbf{x}) = g_{T-1}(\mathbf{x}) + \gamma_T h_T(\mathbf{x}). \quad (9)$$

IV. Attention-based GBM

The idea to apply the attention mechanism to random forests was proposed in [6]. Let us consider how this idea can be adapted to the GBM that is how attention weights can be used in the GBM.

First, we consider the simplest case of the attention weights. This is a way of the direct assignment of non-parametric weights to trees (h_t) without trainable parameter. We also assume that the squared error loss function (5) is used. Then

$$\begin{aligned} g_T(\mathbf{x}) &= h_0(\mathbf{x}) + \sum_{t=1}^{T} \gamma_t h_t(\mathbf{x}) \\ &= h_0(\mathbf{x}) + \sum_{t=1}^{T} \frac{1}{T} \left(\gamma_t \cdot T \cdot h_t(\mathbf{x})\right) \\ &= h_0(\mathbf{x}) + \sum_{t=1}^{T} \omega_t \cdot \hat{h}_t(\mathbf{x}), \end{aligned} \quad (10)$$

where $\omega_t = 1/T$, $\hat{h}_t(\mathbf{x}) = \gamma_t \cdot T \cdot h_t(\mathbf{x})$ is the tree prediction $h_t(\mathbf{x})$ multiplied by γ_t and T.

It should be noted that weights ω_i can be generalized by taking any values satisfying the weight conditions: $\sum_{t=1}^{T} \omega_t = 1$ and $\omega_t \geq 0$, $t = 1, ..., T$. For convenience, we represent $\hat{h}_t$ as a decision tree with the same structure as h_t, but with modified leaf values.

Let us consider now a decision tree as the weak learner in the GBM. Suppose the set $\mathcal{J}_i^{(t)}$ represents indices of instances which fall into the i-th leaf after training the tree at the t-th iteration of GBM. Define the mean vector $\mathbf{A}_t(\mathbf{x})$ and the mean residual value B_t as the mean of training instance vectors, which fall into the i-th leaf of a tree at the t-th iteration, and the corresponding observed mean residual value at the same iteration, respectively, i.e.,

$$\mathbf{A}_t(\mathbf{x}) = \frac{1}{\#\mathcal{J}_i^{(t)}} \sum_{j \in \mathcal{J}_i^{(t)}} \mathbf{x}_j, \quad (11)$$

$$B_t(\mathbf{x}) = \frac{1}{\#\mathcal{J}_i^{(t)}} \sum_{i \in \mathcal{J}_j^{(t)}} \hat{h}_t(\mathbf{x}_j). \quad (12)$$

The distance between $\mathbf{x}$ and $\mathbf{A}_t(\mathbf{x})$ indicates how close the vector $\mathbf{x}$ to vectors $\mathbf{x}_j$ from the dataset D which fall into the same leaf as $\mathbf{x}$. We apply the L_2-norm for the distance definition, i.e., there holds

$$d\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x})\right) = \left\|\mathbf{x} - \mathbf{A}_t(\mathbf{x})\right\|^2. \quad (13)$$

If we return to the Nadaraya-Watson regression model, then the GBM for predicting the target value of $\mathbf{x}$ can be written in terms of the regression model as:

$$G(\mathbf{x}, \mathbf{w}) = h_0(\mathbf{x}) + \sum_{t=1}^{T} \alpha\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x}), \mathbf{w}\right) \cdot B_t(\mathbf{x}). \quad (14)$$

Here $\alpha\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x}), \mathbf{w}\right)$ is the attention weight which is defined in (1) and depends on the mean vector $\mathbf{A}_t(\mathbf{x})$ and on the vector $\mathbf{w}$ of training attention parameters. Here we can say that $B_t(\mathbf{x})$ is the value, $\mathbf{A}_t(\mathbf{x})$ is the key, and $\mathbf{x}$ is the query in terms of the attention mechanism. The weights satisfy the following condition:

$$\sum_{t=1}^{T} \alpha\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x}), \mathbf{w}\right) = 1. \quad (15)$$

The optimal parameters $\mathbf{w}$ can be found by minimizing the expected loss function over a set $\mathcal{W}$ of parameters as follows:

$$\mathbf{w}_{opt} = \arg\min_{\mathbf{w}\in\mathcal{W}} \sum_{j=1}^{n} \left(y_j - G(\mathbf{x}_j, \mathbf{w})\right). \quad (16)$$

The next question is how to define the attention weights $\alpha\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x}), \mathbf{w}\right)$ such that the optimization problem (16) would be simply solved. An efficient way for defining the attention weights has been proposed in [6] where the well-known Huber's ϵ-contamination model [8] was used for trainable parameters $\mathbf{w}$. The Huber's ϵ-contamination model can be represented as

$$(1-\epsilon)\cdot P + \epsilon \cdot Q, \quad (17)$$

where the probability distribution P is contaminated by some arbitrary distribution Q; the rate $\epsilon \in [0,1]$ is a model parameter (contamination parameter) which reflects how "close" we feel that Q must be to P [32].

If we assume that the attention weights $\alpha\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x}), \mathbf{w}\right)$ are expressed through the softmax function $\mathrm{softmax}(d\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x})\right))$ (see (3)), which provides the probability distribution P of the distance $d\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x})\right)$ between vector $\mathbf{x}$ and the mean vector $\mathbf{A}_t(\mathbf{x})$ with some parameters for all iterations, $t = 1, ..., T$, then correction of the distribution can be carried out by means of the ϵ-contamination model, i.e., by means of the probability distribution Q. Suppose that the distribution Q is produced by parameters $\mathbf{w}$ which can take arbitrary values in the T-dimensional unit simplex $\mathcal{W}$, i.e., Q is arbitrary such that $w_1 + ... + w_T = 1$. Then we can write the attention weights as (17), i.e.,

$$\begin{aligned}
&\alpha\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x}), \mathbf{w}\right) \\
&= (1-\epsilon)\cdot P + \epsilon \cdot Q \\
&= (1-\epsilon)\cdot \mathrm{softmax}(d\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x})\right)) + \epsilon \cdot w_t. \quad (18)
\end{aligned}$$

The main advantage of the above representation is that the attention weights linearly depend on the trainable parameters $\mathbf{w}$. We will see below that this representation leads to the quadratic optimization problem for computing optimal weights. This is a very important property of the proposed attention weights because we do not need to numerically solve complex optimization problem for computing the weights. The standard quadratic optimization problem can be only solved, which has a unique solution. The parameter ϵ is the tuning parameter. It is changed from 0 to 1 to get some optimal value which allows us to get the highest regression performance on the validation set. Moreover, the attention weights use the non-trainable softmax function which is simply computed and does not need to be learned.

Substituting (13) into (18), we get the following final form of the attention weights:

$$\begin{aligned}
&\alpha\left(\mathbf{x}, \mathbf{A}_t(\mathbf{x}), \mathbf{w}\right) \\
&= (1-\epsilon)\cdot \mathrm{softmax}\left(\frac{\left\|\mathbf{x} - \mathbf{A}_t(\mathbf{x})\right\|^2}{2}\delta^t\right) + \epsilon \cdot w_t. \quad (19)
\end{aligned}$$

Here $\delta \in [0,1]$ is the discount factor such that δ^t is decreased with the number of iteration t. It determines how much the impact of the attention weight is decreased with the number of iteration t. The discount factor is a tuning parameter.

Hence, the expected loss function for training parameters w can be written as:

$$\min_{\mathbf{w}\in\mathcal{W}} \sum_{s=1}^{n} \left(y_s - h_0(\mathbf{x}) - \sum_{t=1}^{T} F_t(\mathbf{x}_s, \delta^t, \epsilon, w_t)\right)^2, \quad (20)$$

where

$$F_t(\mathbf{x}, \delta^t, \epsilon, w_t) = B_t(\mathbf{x})\left((1-\epsilon)D_t(\mathbf{x}, \delta^t) + \epsilon \cdot w_t\right), \quad (21)$$

$$D_t(\mathbf{x}, \delta^t) = \mathrm{softmax}\left(\frac{\left\|\mathbf{x} - \mathbf{A}_t(\mathbf{x})\right\|^2}{2}\delta^t\right), \quad (22)$$

Problem (20) is the standard quadratic optimization problem with linear constraints $\mathbf{w} \in \mathcal{W}$, i.e., $w_t \geq 0$, $t = 1, ..., T$, and $\sum_{t=1}^{T} w_t = 1$.

As a result, we get a simple quadratic optimization problem whose solution does not meet any difficulties. from the computational point of view because its training is based on solving the standard quadratic optimization problem.

V. Numerical experiments

The attention-based GBM is evaluated and investigated solving regression problems on 11 datasets from open sources. Dataset Diabetes can be found in the corresponding R Packages. Three datasets Friedman 1, 2 3 are described at site: https://www.stat.berkeley.edu/~breiman/bagging.pdf. Datasets Regression and Sparse are available in package "Scikit-Learn". Datasets Wine Red, Boston Housing, Concrete, Yacht Hydrodynamics, Airfoil are taken from UCI Machine Learning Repository [33]. A brief introduction about these data sets is represented in Table I where m and n are numbers of features and instances, respectively. A more detailed information is available from the above resources.

TABLE I. A brief introduction about the regression data sets

Data set	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

We use the coefficient of determination denoted R^2 and the mean absolute error (MAE) for the regression evaluation. The greater the value of the coefficient of determination and the smaller the MAE, the better results we get. Every GBM has 200 iterations. Decision trees at each iteration are built such

that at least 10 instances fall into every leaf of trees. This condition is used to get desirable estimates of vectors $\mathbf{A}_t(\mathbf{x}_s)$.

To evaluate the average accuracy measures, we perform a cross-validation with 100 repetitions, where in each run, we randomly select $n_{\text{tr}} = 4n/5$ training data and $n_{\text{test}} = n/5$ testing data. The best results in all tables are shown in bold.

In all tables, we compare R^2 and the MAE for three cases: (**GBM**) the GBM without the softmax and without attention model; (**Non-param**) a special case of the AGBoost model when trainable parameters $\mathbf{w}$ are not learned, and they are equal to $1/T$; (**AGBoost**) the proposed AGBoost model with trainable parameters $\mathbf{w}$.

The optimal values of the contamination parameter ϵ_{opt} and the discount factor δ_{opt} are provided. The case $\epsilon_{opt} = 1$ means that the attention weights are totally determined by the tree weights and do not depend on each instance. The case $\epsilon_{opt} = 0$ means that weights of trees are determined only by the softmax function without trainable parameters.

Measures R^2 and MAE for three cases (GBM, Non-param and AGBoost) are shown in Table II with the original decision trees as base learners. It can be seen from Table II that the proposed AGBoost model outperforms the GBM itself and the non-parametric model or is comparable with these models for all datasets.

To formally show the outperformance of the proposed AGBoost model with the original decision trees as base learners, we apply the t-test which has been proposed and described by Demsar [34] for testing whether the average difference in the performance of two models, AGBoost and GBM, is significantly different from zero. Since we use differences between accuracy measures of AGBoost and GBM, then they are compared with 0. The t statistics in this case is distributed according to the Student distribution with $11 - 1$ degrees of freedom. Results of computing the t statistics of the difference are the p-values denoted as p and the 95% confidence interval for the mean 0.024, which are $p = 0.0013$ and $[0.012, 0.037]$, respectively. The t-test demonstrates the outperformance of AGBoost in comparison with the GBM because $p < 0.05$. We also compare AGBoost with the non-parametric model. We get the 95% confidence interval for the mean 0.018, which are $p = 0.0045$ and $[0.007, 0.029]$, respectively. Results of the second test also demonstrate the outperformance of AGBoost in comparison with the non-parametric model.

Measures R^2 and MAE for three cases (GBM, Non-param and AGBoost) are shown in Table III with the ERT as base learners. It can be seen from Table III that the proposed AGBoost model outperforms the GBM itself and the non-parametric model for all datasets.

To formally show the outperformance of the proposed AGBoost model with ERTs as base learners, we again apply the t-test. Results of computing the t statistics are the p-values and the 95% confidence interval for the mean 0.067, which are $p = 0.0012$ and $[0.033, 0.100]$, respectively. The t-test demonstrates the clear outperformance of AGBoost in comparison with the GBM. We also compare AGBoost with the non-parametric model. We get the 95% confidence interval for the mean 0.062, which are $p = 0.0019$ and $[0.029, 0.095]$, respectively. Results of the second test also demonstrate the outperformance of AGBoost in comparison with the non-parametric model. It follows from the obtained results that models with the ERTs as base learners provide better results than the models with original decision trees.

VI. Concluding remarks

A new model of the attention-based GBM has been proposed. It can be regarded as an extension of the attention-based random forest [6]. The proposed model inherits advantages of the attention mechanism and the GBM. Moreover, it allows us to avoid using neural networks. Numerical experiments have demonstrated that incorporating the attention model into the GBM improves the original GBM.

At the same time, the proposed model is rather flexible and this fact allows us to determine several directions for further research. First, we have investigated only the Huber's ϵ-contamination model for incorporating the trainable parameter into the attention. However, there exist some statistical models which have similar properties. Their use and study instead of the Huber's contamination model is a direction for further research. The proposed AGBoost model uses non-parametric softmax function for computing the attention weights. It is interesting to extend the proposed model to the case of the parametric softmax function with trainable parameters. It should be noted that additional trainable parameters in the softmax may significantly complicate the model. However, efficient computation algorithms are also directions for further research. It should be also noted that the proposed attention-based approach can be incorporated into other GBM models, for example, into XGBoost, pGBRT, SGB, etc. This is also a direction for further research.

Acknowledgement

The research results have been obtained in December of 2021. This work is supported by the Russian Science Foundation under grant 21-11-00116.

References

[1] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, 2021.
[2] S. Chaudhari, V. Mithal, G. Polatkan, and R. Ramanath, "An attentive survey of attention models," Apr 2019, arXiv:1904.02874.
[3] A. Correia and E. Colombini, "Attention, please! A survey of neural attention models in deep learning," Mar 2021, arXiv:2103.16775.
[4] ——, "Neural attention models in deep learning: Survey and taxonomy," Dec 2021, arXiv:2112.05909.
[5] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," Jul 2021, arXiv:2106.04554.
[6] L. Utkin and A. Konstantinov, "Attention-based random forest and contamination model," Jan 2022, arXiv:2201.02880.
[7] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
[8] P. Huber, *Robust Statistics*. New York: Wiley, 1981.
[9] J. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, pp. 1189–1232, 2001.
[10] ——, "Stochastic gradient boosting," *Computational statistics & data analysis*, vol. 38, no. 4, pp. 367–378, 2002.
[11] P. Buhlmann and T. Hothorn, "Boosting algorithms: Regularization, prediction and model fitting," *Statistical Science*, vol. 22, no. 4, pp. 477–505, 2007.

TABLE II. MEASURES R^2 AND MAE FOR COMPARISON OF THREE MODELS BY OPTIMAL VALUES OF CONTAMINATION PARAMETER AND DISCOUNT FACTOR WITH THE ORIGINAL DECISION TREES AS BASE LEARNERS

			R^2			MAE		
Data set	ϵ_{opt}	δ_{opt}	GBM	Non-param	AGBoost	GBM	Non-param	AGBoost
Diabetes	0	0.9	0.390	**0.391**	**0.391**	**46.11**	46.53	46.53
Friedman 1	1	0.01	0.582	0.582	**0.595**	2.269	2.269	**2.219**
Friedman 2	0.778	1	0.909	0.913	**0.944**	85.39	83.79	**62.21**
Friedman 3	0.667	0.9	0.696	0.700	**0.715**	0.135	0.136	**0.125**
Regression	0.778	1	0.572	0.577	**0.626**	88.34	86.91	**82.00**
Sparse	0.000	1	0.598	**0.651**	**0.651**	1.646	**1.582**	**1.582**
Airfoil	0.889	1	0.792	0.794	**0.831**	2.456	2.443	**2.156**
Boston	0.667	1	0.832	0.838	**0.845**	2.481	2.465	**2.379**
Concrete	0.889	1	0.847	0.846	**0.872**	4.970	4.981	**4.382**
Wine	0.111	0.5	0.412	0.412	**0.413**	0.475	0.475	**0.470**
Yacht	1	0.01	0.973	0.973	**0.991**	1.594	1.594	**0.675**

TABLE III. MEASURES R^2 AND MAE FOR COMPARISON OF THREE MODELS BY OPTIMAL VALUES OF CONTAMINATION PARAMETER AND DISCOUNT FACTOR WITH THE ERT AS BASE LEARNERS

			R^2			MAE		
Data set	ϵ_{opt}	δ_{opt}	GBM	Non-param	AGBoost	GBM	Non-param	AGBoost
Diabetes	0.111	0.01	0.435	0.434	**0.440**	44.80	44.83	**44.11**
Friedman 1	1	0.01	0.561	0.561	**0.612**	2.248	2.248	**2.146**
Friedman 2	1	0.01	0.874	0.874	**0.975**	98.57	98.57	**41.12**
Friedman 3	0.667	1	0.637	0.639	**0.800**	0.160	0.158	**0.110**
Regression	0.778	0.9	0.495	0.487	**0.604**	96.88	97.76	**83.30**
Sparse	0.556	1	0.558	0.611	**0.659**	1.777	1.656	**1.504**
Airfoil	1	0.01	0.747	0.747	**0.835**	2.734	2.734	**2.151**
Boston	0.778	1	0.835	0.837	**0.868**	2.463	2.457	**2.260**
Concrete	1	0.01	0.817	0.817	**0.867**	5.577	5.577	**4.565**
Wine	0.778	1	0.396	0.400	**0.409**	0.484	0.481	**0.468**
Yacht	1	0.01	0.973	0.973	**0.992**	1.596	1.596	**0.642**

[12] A. Zhang, Z. Lipton, M. Li, and A. Smola, "Dive into deep learning," *arXiv:2106.11342*, Jun 2021.

[13] E. Nadaraya, "On estimating regression," *Theory of Probability & Its Applications*, vol. 9, no. 1, pp. 141–142, 1964.

[14] G. Watson, "Smooth regression analysis," *Sankhya: The Indian Journal of Statistics, Series A*, pp. 359–372, 1964.

[15] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine learning*, vol. 63, pp. 3–42, 2006.

[16] F. Liu, X. Huang, Y. Chen, and J. Suykens, "Random features for kernel approximation: A survey on algorithms, theory, and beyond," Jul 2021, arXiv:2004.11154v5.

[17] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. Colwell, and A. Weller, "Rethinking attention with performers," in *2021 International Conference on Learning Representations*, 2021.

[18] X. Ma, X. Kong, S. Wang, C. Zhou, J. May, H. Ma, and L. Zettlemoyer, "Luna: Linear unified nested attention," Nov 2021, arXiv:2106.01540.

[19] H. Peng, N. Pappas, D. Yogatama, R. Schwartz, N. Smith, and L. Kong, "Random feature attention," in *International Conference on Learning Representations (ICLR 2021)*, 2021, pp. 1–19.

[20] I. Schlag, K. Irie, and J. Schmidhuber, "Linear transformers are secretly fast weight programmers," in *International Conference on Machine Learning 2021*. PMLR, 2021, pp. 9355–9366.

[21] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in neurorobotics*, vol. 7, no. Article 21, pp. 1–21, 2013.

[22] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY: ACM, 2016, pp. 785–794.

[23] S. Tyree, K. Weinberger, K. Agrawal, and J. Paykin, "Parallel boosted

[24] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to regression trees for web search ranking," in *Proceedings of the 20th international conference on World wide web*. ACM, 2011, pp. 387–396.
deep neural networks," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*. Melbourne, Australia: AAAI Press, 2017, pp. 3553–3559.

[25] J. Feng, Y. Yu, and Z.-H. Zhou, "Multi-layered gradient boosting decision trees," in *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018, pp. 3551–3561.

[26] J. Feng, Y. Xu, Y. Jiang, and Z.-H. Zhou, "Soft gradient boosting machine," Jun. 2020, arXiv:2006.04059.

[27] A. Konstantinov and L. Utkin, "A generalized stacking for implementing ensembles of gradient boosting machines," Oct. 2020, arXiv:2010.06026.

[28] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," Sep 2014, arXiv:1409.0473.

[29] T. Luong, H. Pham, and C. Manning, "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, 2015, pp. 1412–1421.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, 2017, pp. 5998–6008.

[31] A. Smola and B. Scholkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004.

[32] J. Berger, *Statistical Decision Theory and Bayesian Analysis*. New York: Springer-Verlag, 1985.

[33] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[34] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.