

# A Bi-level Evolutionary Model Tree Induction Approach for Regression

Safa Mahouachi<sup>1</sup>, Maha Elarbi<sup>1</sup>, Khaled Sethom<sup>1</sup>, Slim Bechikh<sup>1</sup>, and Carlos A. Coello Coello<sup>2</sup>

<sup>1</sup>SMART Lab, ISG, University of Tunis, Tunisia

<sup>2</sup>Departamento de Computación, CINVESTAV-IPN, Mexico City, Mexico

**Abstract**—Supervised machine learning techniques include classification and regression. In regression, the objective is to map a real-valued output to a set of input features. The main challenge that existing methods for regression encounter is how to maintain an accuracy-simplicity balance. Since Regression Trees (RTs) are simple to interpret, many existing works have focused on proposing RT and Model Tree (MT) induction algorithms. MTs are RTs with a linear function at the leaf nodes rather than a numerical value are able to describe the relationship between the inputs and the output. Traditional RT induction algorithms are based on a top-down strategy which often leads to a local optimal solution. Other global approaches based on Evolutionary Algorithms (EAs) have been proposed to induce RTs but they can require an important calculation time which may affect the convergence of the algorithm to the solution. In this paper, we introduce a novel approach called Bi-level Evolutionary Model Tree Induction algorithm for regression, that we call BEMTI, and which is able to induce an MT in a bi-level design using an EA. The upper-level evolves a set of MTs using genetic operators while the lower-level optimizes the Linear Models (LMs) at the leaf nodes of each MT in order to fairly and precisely compute their fitness and obtain the optimal MT. The experimental study confirms the outperformance of our BEMTI compared to six existing tree induction algorithms on nineteen datasets.

**Index Terms**—Model Trees, Induction, Regression, Bi-level optimization, Evolutionary Algorithm.

## I. INTRODUCTION

Machine Learning (ML) is a sub-area of Artificial Intelligence (AI) where computers learn to build a model upon historical data in order to be able to make a prediction on new data. Supervised ML tasks aim to define a function or a model that maps input feature vectors to output vectors and are divided into two categories: regression and classification [1]. In regression tasks, the outputs are real values (i.e., an integer or a floating-point value). Several regression techniques exist to solve regression problems such as Artificial Neural Networks (ANNs), Support Vector Machines (SVMs), Naïve Bayes (NB), and Regression Trees (RTs) [2]. Tree-based techniques are widely used in many real-world problems since they provide compact and interpretable representations as opposed to other “black-box” methods, for instance SVMs and ANNs [3] which are complex and computationally expensive. Among the existing top-down RT induction algorithms in the literature we can cite the Classification And Regression Tree (CART) [4] which is based on recursive partitioning, and the REPTree [5] which consists of building a tree based on the variance and then applies a tree pruning procedure. Other tree

induction algorithms use Model Trees (MTs) instead of RTs such as the M5 system [6] which builds MTs with multi-variate linear models at the leaves, as well as an ensemble-based algorithm using M5 called Ensemble of M5 (E-M5) [7]. MTs can be thought of as a variation of RTs in which each leaf node is represented by a regression function rather than a numerical value [8]. The regression function in the leaves of an MT can model the relationship between the input variables and the predicted value. All of the aforementioned algorithms are based on a top-down strategy and are considered greedy and lead generally to locally optimal solutions. Recently, global tree induction approaches based on Evolutionary Algorithms (EAs) have been proposed to induce Decision Trees (DTs) and RTs and are popular alternatives to traditional top-down algorithms. Fan and Gray [9] proposed the TARGET algorithm for RT induction. Czajkowski and Kretowski [10] proposed the Global Model Tree (GMT) framework followed by all its extensions, i.e., oGMT [11], mGMT [8], and pGMT [3]. Geerts et al. [12] proposed the GeoTree algorithm to induce RTs with three split types. Evolutionary Algorithms (EAs) have been popular in decision and regression tasks since they provide a global search and can avoid being stuck in local optima [14].

One key issue of the top-down induction algorithms is that they are greedy and based on recursive partitioning which may lead to obtain locally optimal solutions. Furthermore, although EAs perform a global exploration of the search space and could generate more efficient solutions, they might increase the calculation time of the algorithm used at each generation due to a poor evaluation of the individuals which may affect the convergence of the algorithm to the optimal solution [3]. Motivated by these observed issues, we propose a new approach to induce MTs based on a bi-level design called Bi-level Evolutionary Model Tree Induction (BEMTI) for regression. BEMTI solves a regression problem by evolving and evaluating a population of induced MTs in the upper-level, and optimizing the Linear Models (LMs) at the leaves of the MTs, in the lower-level. The use of an MT as a base regressor in BEMTI ensures the explainability of the prediction model whereas the bi-level optimization guarantees a fair and precise evaluation. The main contributions of this paper can be summarized as follows:

- 1) Considering the regression task as a bi-level optimization

tion problem where we induce, evolve and evaluate a set of MTs in the upper-level, and a sequence of LMs of each MT at the lower-level.

- 2) Introducing a Bi-level Evolutionary Model Tree Induction algorithm, that we call BEMTI, as a solution to a regression problem. MTs ensure the interpretability of the model whereas the evolutionary bi-level strategy guarantees a fair and precise evaluation of the solutions.
- 3) Showing that BEMTI is able to obtain better results in terms of prediction error than several tree induction algorithms from the literature when applied on regression datasets, thus confirming the added value of BEMTI in solving regression problems compared to greedy approaches and recent EAs.

The remainder of this paper is organized as follows. Section II provides a brief review on existing tree induction algorithms. Section III describes our proposed approach called BEMTI. Section IV presents the experimental environment. In Section V the results of the experiments are reported and discussed. Finally, in Section VI, we conclude the paper and present some possible paths for future work in this area.

## II. PREVIOUS RELATED WORK

### A. Traditional machine learning methods for regression

Regression tasks aim to find a function that maps a vector of independent feature variables  $X = (x_1, x_2, \dots, x_i, \dots, x_d)$  to a dependant real-valued output variable  $y$ , for a given set of training data, where  $d$  is the number of features. ML algorithms for regression can be classified into the following categories: ANNs, SVMs, NB, and tree-based methods [2]. Wu and Yen [15] proposed to use an ANN for regression by training a network to find a regression function that maps the inputs into the output in the observed data. ANNs have the advantage of being able to model non-linear relationships between the features and the target in a regression task [2]. However, because of their “black-box” structure, the training process is incomprehensible for humans which makes ANN models for regression hard to understand and interpret [16]. Drucker et al. [17] proposed to use SVMs for regression and introduced Support Vector Regression (SVR). The basic idea behind SVR is to find the hyperplane that has the maximum number of points of the data. Frank et al. [18] proposed to apply the probabilistic classifier NB to solve regression tasks. Existing works showed that NB has the same performance as linear regression and worse than tree-based algorithms. RTs offer interpretable and effective models having a comprehensible decision making process as opposed to “black-box” algorithms [3], [11], [16]. One of the most prominent DT and RT induction algorithm is the Classification And Regression Tree (CART) system introduced by Breiman et al. [4]. The algorithm consists of constructing RTs by iterative splitting of the data where each split minimizes the Residuals Sum of Squares (RSS). Other tree induction algorithms employ MTs which are variants of RTs where single values at the leaves are replaced by multi-variate linear functions [3], [19], and

allow more accurate predictions than RTs. Among the MTs induction algorithm we refer to the M5 system proposed by Quinlan [6] which generates MTs by performing a splitting process based on the standard deviation of the node involved in the subset of the training data and then performs a pruning process, and to the Stepwise Model Tree Induction (SMOTI) proposed by Malerba et al. [20] and which induces MTs in a stepwise fashion. Additionally, Arora et al. [5] proposed the REPTree algorithm which induces an RT with a top-down strategy by varying the variance either high or low and applying a reduced-error pruning process. Other algorithms based on ensemble learning have been proposed to induce RTs and consist of dividing the training data and building multiple models and then averaging their results [21]. Breiman [22] proposed the Bagging (BAG) method, whose name comes from the contraction of the words Bootstrap and Aggregating. BAG constructs an ensemble of trees independently using bootstrapping of the data and aggregating the results in the end. Boosting [23] is another ensemble method which differs from bagging since each sample is drawn according to the performance of the basic rule applied to the previous sample. Later on, Breiman [24] proposed random forests which selects a random subset of features to split each node in a tree. Sattari et al. [7] proposed the E-M5 to induce MTs using four ensemble-based approaches along with the M5 algorithm as a base regressor which are: stochastic gradient boosting, bagging, rotation forest, and random sub-space. The main challenges of the top-down tree induction algorithms are their greedy nature without a global consideration of further nodes in the tree and that they generally lead to local optima [9]. Moreover, ensemble approaches could lead to “black-box” models which are difficult to interpret [7].

### B. Evolutionary methods for regression

More recent studies have proposed global tree induction methods based on EAs as an alternative to classical top-down tree induction algorithms. EA-based methods could enhance the chances of converging to global optima and avoid locally optimal solutions which are sensitive to small changes in the data. Fan and Gray [9] proposed TARGET (Tree Analysis with Randomly Generated and Evolved Trees), which is based on a genetic algorithm to search for the optimal solution. TARGET begins by creating randomly a population of RTs using a recursive split. Each node could be subject to a split based on a split probability; otherwise, it becomes a leaf node. Genetic operators are then applied and the best trees are retained from one generation to another using the Bayesian Information Criterion (BIC) as a measure of tree fitness. Czajkowski and Kretowski proposed the GMT framework [10] to induce regression trees by applying an EA. The initial population in GMT is composed by MTs generated randomly using a top-down strategy like M5 or CART. MTs are evaluated using the BIC and the algorithm stops when some convergence criteria are satisfied. In some further work, Czajkowski and Kretowski [11] extended the GMT solution with the use of a linear combination of features in the split nodes instead of uni-variate

TABLE I  
DESCRIPTION OF EXISTING STATE-OF-THE-ART REGRESSION TREE INDUCTION ALGORITHMS.

Reference	Algorithm	Category		Solution encoding	Limitations
		Greedy	EA		
[4]	CART	✓		Builds RTs using recursive partitioning of the data into sub-groups based on split rules selected in a forward stepwise search.	No consideration of nodes further down the tree and leads to local optimal solutions.
[6]	M5	✓		Builds MTs with multi-variate linear models at each node.	No consideration of nodes further down the tree and leads to local optimal solutions.
[20]	SMOTI	✓		Constructs MTs stepwise where each node can be a split or a straight line regression to ensure a global effect.	Generates MTs that may overfit the data.
[5]	REPTree	✓		Builds a decision or regression tree based on the variance and then applies a reduced error pruning.	Produces large trees which may affect the interpretability of the model.
[22]	BAG	✓		Builds multiple trees based on a subset of the training data and then aggregates the results.	Affects the interpretability of the model.
[7]	E-M5	✓		Builds model trees using ensemble methods based on M5.	Generates complex models difficult to interpret.
[9]	TARGET		✓	Repeatedly and randomly generates trees where each node can be a split node or a leaf node based on a fixed probability until no more nodes can be split.	Evolves regression trees.
[10]	GMT		✓	Induces MTs which globally search for the best tree structure, with tests at internal nodes and LMs at the leaves.	Does not include the oblique splits in internal nodes and the tree is larger.
[11]	oGMT		✓	Induces oblique MTs by splitting hyper-planes in non-terminal nodes and multiple LMs in the leaves.	Increases the calculation time.
[8]	mGMT		✓	Induces trees with uni-variate, oblique, regression, or model representations.	Not efficient for large datasets, does not include non-linear regression models in the leaves.
[3]	pGMT		✓	Induces MTs using a Pareto-based strategy with a multi-objective fitness function.	Increases the calculation time of each evolutionary loop and may affect the convergence of the EA.
[12]	GeoTree		✓	Induces multi-variate decision trees with axis-parallel splits, oblique splits or Gaussian splits. Uses an EA for splits search.	Does not perform a global search and generates near-optimal solutions.

tests and introduced the Oblique Global Model Tree algorithm (oGMT) where the generated trees are partitioned by oblique hyper-planes and are much smaller and more accurate than the axis-aligned tree. The same authors expanded their GMT approach in the Mixed Global Model Tree (mGMT) algorithm [8] where the structure of the induced tree is not known in advance (i.e., uni-variate, oblique, regression, model). This way, the solution is able to self-adapt to the analyzed data of the problem. A later version of the GMT framework called Pareto approach for GMT (pGMT) [3] aims to find Pareto efficient solutions by adopting a Pareto-based multi-objective optimization strategy in the fitness function employed in GMT. Geerts et al. [12] proposed the Geospatial Regression Tree (GeoTree) algorithm which induces regression trees with three types of splits: axis-parallel, oblique and Gaussian. GeoTree employs a genetic algorithm to generate the best orthogonal, oblique and Gaussian candidate splits. Table I summarizes the reviewed tree induction algorithms and underlines for each one its category (i.e., greedy or EA), its solution encoding and its main limitations.

Although the aforementioned studies have proposed several RT induction algorithms, three main issues persist. First, the statistical and ML approaches generate local-optimal solutions (i.e., local-optimal models) due to their greedy search nature. Second, the lack of maintaining the interpretability-accuracy trade-off is the main barrier of the applicability of

deep learning-based methods. Finally, the main shortcoming of EAs concerns the convergence of the algorithm to the optimal solution caused by the interruption of the evolution when all the individuals become identical and no relevant exploration can be performed. This can be explained because the proposed EAs consider the induction task as a single-level optimization problem. Therefore, we propose a novel approach called BEMTI which employs an EA to induce MTs in a bi-level fashion as a solution to the regression problem.

### III. OUR PROPOSED APPROACH: BEMTI

#### A. Main idea and motivation

Similarly to existing EAs for regression, the GMT framework considers the induction task as a single-level optimization problem [10]. Each individual of the initial population, (i.e., MT), is induced using a traditional top-down strategy algorithm. Internal nodes define a splitting condition whereas leaf nodes correspond to linear or local models (LMs) represented by a multi-variate regression function. Each LM in the MT is a linear weighted sum of variables augmented by a bias value. However, before evaluating a particular LM, its parameters should be optimized to come up with a more precise and fair fitness value computation. Motivated by this observation, the main idea of our approach (BEMTI) consists of modelling the MT induction task as a Bi-Level Optimization Problem (BLOP) [25] and then solve it using a suitable bi-level

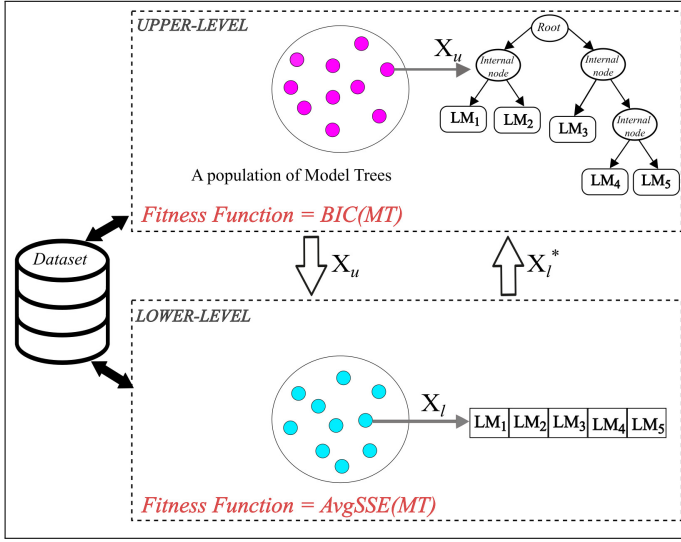


Fig. 1. Illustration of the main principle of our BEMTI approach.

EA. Evolutionary optimization allows escaping local optima as opposed to traditional top-down induction algorithms while MTs ensure the interpretability of the results since they are able to model the relationship between input features and the output. Fig. 1 illustrates the main framework of our proposed BEMTI approach. The upper-level is composed by a population of induced MTs and then, a new population of solutions is generated over time using genetic operators, (i.e., crossover and mutation operators). A lower-level optimization process, involving the coefficients of the sequence of LMs, which constitute the lower-level solution  $X_l$ , is carried out for every upper-level solution  $X_u$ . Once again, the genetic information of the lower-level population is recombined and mutated to form the new population. The individuals of the lower-level population are computed using the  $AvgSSE(MT)$  defined in equation (7). When the algorithm reaches a stopping criterion, the sequence of the best LMs' parameter vectors, i.e.,  $X_l^*$ , is sent to the upper-level in order to precisely and fairly measure the quality of this MT using the BIC defined in equation (1) as the fitness function corresponding to each solution.

### B. Upper-level

1) *Solutions encoding*: The upper-level population is initialized using a top-down strategy to induce the MTs. Each upper-level solution is encoded as an mGMT since the representation of the model could affect its predictive performance [8]. In fact, the structure of the nodes in a tree-shaped model (decision, regression, uni-variate regression, multi-variate regression), allows to obtain different kinds of trees (decision tree, regression tree, axis-parallel tree, oblique tree). However, in real-life problems, it is difficult to fix the most appropriate representation in advance [8]. The mGMT induces trees aligned with the characteristics of the data. The split nodes of the MT can be represented by uni-variate and multi-variate conditions. The first involves a single variable that splits data

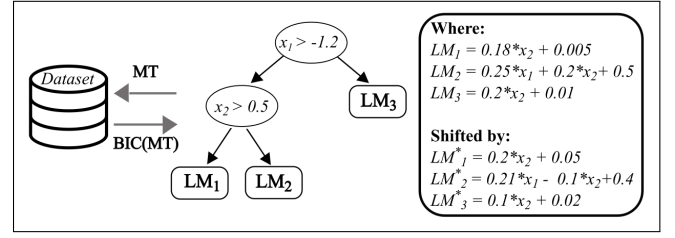


Fig. 2. Illustration of the upper-level individual of our proposed BEMTI approach.

space in an axis-parallel fashion, while the second combines several variables so that the resulting splitting hyperplane would be oblique. Fig. 2 illustrates an individual of the upper-level where each internal node is represented by a uni-variate condition. The leaf nodes are represented by multi-variate LMs. In BEMTI, we limit the maximum number of variables of each oblique split to three in order to effectively capture the data distribution [11]. We note that the condition sign is less-or-equal ( $\leq$ ) and the variables' weights lie within  $[-10, 10]$  following the recommendations of Czajkowski and Kretowski [26].

2) *Fitness function*: The upper-level solutions of BEMTI are evaluated based on the quality of each MT. We adopt the BIC as the fitness function of the upper-level since it allows evaluating both effectiveness and efficiency. The former is measured in terms of the MT error, while the latter assesses the tree complexity. The BIC is expressed as follows [3]:

$$BIC(MT) = Error(MT) + Complexity(MT) \quad (1)$$

where

$$Error(MT) = -2\ln(L(MT)) \quad (2)$$

and

$$Complexity(MT) = \ln(n) * k(MT) \quad (3)$$

$L(MT)$  is the maximum likelihood function of the  $MT$  and is expressed as follows [3]:

$$\ln(L(MT)) = -0.5n * [\ln(2\pi) + \ln(SSE(MT)/n) + 1] \quad (4)$$

where  $n$  is the number of instances in the dataset,  $SSE(MT)$  is the sum of squared residuals of the tree  $MT$  and is defined as follows [27]:

$$SSE(MT) = \sum_{i=1}^n (y - \hat{y})^2 \quad (5)$$

where  $y$  is the real value of the output of the dataset and  $\hat{y}$  is the predicted one.  $k(MT)$  estimates the number of the  $MT$  parameters as follows [10]:

$$k(MT) = 2(Q(MT) + M(MT)) \quad (6)$$

where  $Q(MT)$  is the number of nodes in the  $MT$  and  $M(MT)$  is the sum of the numbers of attributes of all leaf nodes' LMs in the tree  $MT$ .

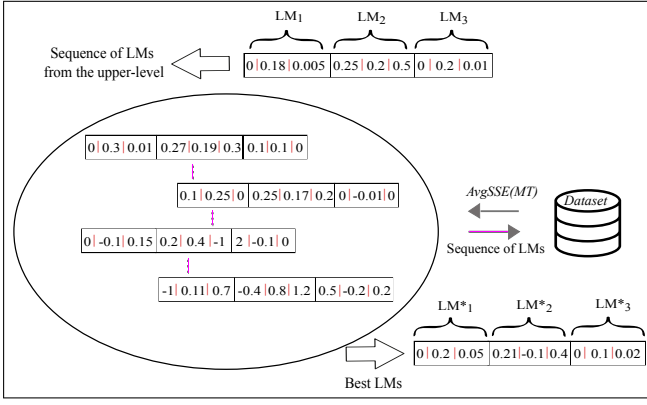


Fig. 3. Illustration of the lower-level individual of BEMTI.

3) *Genetic operators*: For both levels, we use the binary tournament operator [28] for mating selection. At each iteration, this operator randomly chooses two solutions and then preserves only the best one in the mating pool. This process is repeated until filling this pool. For the crossover operator, we adopt the subtree exchange [29] that first selects an internal node in each parent tree and then switches the two resulting subtrees. As for the mutation operator, we can vary either the condition of an internal node or the LMs at the leaves. We employ the Polynomial-based Mutation (PM) operator [30] to vary the attributes, the weights, and/or the related threshold of an internal node. The operators and variables of the LMs in the leaf nodes can be randomly replaced by other ones while the coefficients are varied using the PM operator. The latter is applied with a repair strategy that allows respecting the definition intervals of the coefficients.

### C. Lower-level

1) *Solution encoding*: The lower-level of our approach BEMTI is composed of a population of LMs where each LM is represented by a linear weighted sum of variables augmented by a bias value. As shown in Fig. 3, each individual of the lower-level can be seen as a sequence of the coefficients of a set of LMs. Based on the recommendations of previous works [10], [26], the coefficients vary within the interval  $[-10, 10]$ , while the bias is generated from the normal distribution  $N(0, \sigma^2)$  [31].

2) *Fitness function*: For each upper-level solution MT, a vector composed of the coefficients of the set of LMs is passed to the lower-level and will be subject to an optimization process. The candidate solutions of the lower-level are computed using the fitness function defined as:

$$AvgSSE(MT) = \frac{1}{NLM(MT)} \sum_{i=1}^{NLM(MT)} SSE_i(MT) \quad (7)$$

where  $NLM(MT)$  is the number of the LMs of the tree  $MT$  and  $SSE_i(MT)$  is the  $SSE$  value of the  $i^{th}$  LM of  $MT$ .

3) *Genetic operators*: Since the lower-level solution is encoded as a vector of real numbers composed of a sequence of weights and biases, we use the Simulated Binary Crossover

TABLE II  
CHARACTERISTICS OF THE ANALYZED DATASETS.

Name	#Features	#Instances
2D planes	10	40768
Ailerons	40	13750
Bank32NH	32	8192
Bank8FM	8	8192
California Housing	8	20640
CompAct	22	8192
CompAct(s)	8	8192
Delta Ailerons	6	7129
Delta Elevators	6	9517
Elevators	18	16559
Fried	10	40768
House 16H	16	22784
House 8L	8	22784
Kinematics	8	8192
Pole Telecomm	48	15000
Puma32H	32	8192
Puma8NH	8	8192
Stock	10	950
Wisconsin Cancer	32	194

(SBX) operator [32], which is one of the most popular crossover operators adopted in numerical optimization (as well as the PM operator) to vary the coefficients of the LMs.

## IV. EXPERIMENTAL STUDY

### A. Benchmark datasets

Several regression datasets used in the original papers of the peer algorithms were analyzed in order to conduct the comparative study of our BEMTI against six RT and MT induction algorithms. The datasets are detailed in Table II, which shows for each of the nineteen datasets its name (Name), the number of its features (#Features), as well as the number of instances (#Instances). These datasets are available on GitHub at: [github.com/renatopp/arff-datasets/tree/master/regression](https://github.com/renatopp/arff-datasets/tree/master/regression).

### B. Baseline approaches

We conducted our experimental study with a comparison of our proposed BEMTI against the most representative tree induction algorithms from the state-of-the-art which are:

- REPTree [5]: A popular top-down induction algorithm.
- BAG [22]: Ensemble of regression trees allowing to build a large number of predictors based on the CART algorithm.
- M5 [6]: Uni-variate MT inducer which is the greedy counterpart of the GMT framework.
- E-M5 [7]: Ensemble of MTs based on greedy M5.
- mGMT [8]: Global MT inducer using EA, the single-level counterpart of our BEMTI, and
- pGMT [3]: A Pareto-based extension of the GMT framework.

### C. Performance metrics

To assess the performance of the studied algorithms, we used two scale-dependant metrics which are the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE)

TABLE III  
PARAMETERS SETTINGS.

Algorithm	Parameter	Value
<i>Evolutionary-based algorithms</i>		
BEMTI	Tree depth	8
	Upper-level population size	50
	Lower-level population size	100
	Upper-level generations	200
	Lower-level generations	200
	Crossover rate	0.2
	Mutation rate	0.8
	Elitism size	5%
pGMT	Population size	50
	Number of generations	1000
	Crossover rate	0.2
	Mutation rate	0.8
	Elitism size	50%
mGMT	Population size	50
	Number of generations	1000
	Crossover rate	0.2
	Mutation rate	0.8
	Elitism size	2%
<i>Greedy algorithms</i>		
E-M5	Shrinkage rate	1
M5	Split threshold	5%
BAG	Grow and prune ratios	90%-10%
	Bootstrap replicates	25
REPTree	Minimum variance for split	1E-3

since they are considered as the most commonly adopted metrics for regression problems. MAE measures the average of the absolute difference between the actual and the predicted values, it is very easy to compute and to understand and it is relatively robust to outliers. MAE is expressed as follows [33]:

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (8)$$

where  $y_i$  and  $\hat{y}_i$  are respectively the real value and the predicted value of the output of the instance  $i$ , and  $n$  is the number of instances. RMSE is the square root of the Mean Squared Error (MSE) and it is a scale-dependant metric. Nevertheless, its main shortcoming is its sensitivity to forecasting outliers since it is based on squared values of the prediction error. RMSE is defined by the following expression [33]:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (9)$$

#### D. Parameters settings and statistical testing

To assess the performance of our BEMTI, it is important to choose the best configuration of parameters. A parameter tuning process has been conducted to find the most efficient values of the population size and the tree depth for our BEMTI algorithm. The performance of each parameter is evaluated with respect to the RMSE metric tested on the benchmark

TABLE IV  
THE AVERAGE RMSE VALUES OBTAINED BY EACH OF THE CONSIDERED ALGORITHMS OVER 31 RUNS FOR EACH DATASET.

Dataset	REPTree	BAG	M5	E-M5	mGMT	pGMT	BEMTI
2D planes	1.045	1.040	0.996	0.995	0.996	0.995	<b>0.990</b>
Ailerons	2.0E-4	1.8E-4	<b>1.6E-4</b>	<b>1.6E-4</b>	<b>1.6E-4</b>	<b>1.6E-4</b>	<b>1.6E-4</b>
Bank32NH	0.094	0.087	0.082	0.082	0.083	<b>0.081</b>	0.082
Bank8FM	0.040	0.033	0.030	0.029	0.029	<b>0.028</b>	<b>0.028</b>
California Housing	9.7E4	7.8E4	11.3E4	7.7E4	7.4E4	7.3E4	<b>7.1E4</b>
CompAct	3.294	2.794	2.655	2.654	2.654	2.650	<b>2.642</b>
CompAct(s)	3.945	<b>3.220</b>	3.284	3.222	3.263	<b>3.220</b>	3.222
Delta Ailerons	1.8E-4	<b>1.7E-4</b>	<b>1.7E-4</b>	<b>1.7E-4</b>	<b>1.7E-4</b>	<b>1.7E-4</b>	<b>1.7E-4</b>
Delta Elevators	1.4E-3	1.4E-3	1.4E-3	1.4E-3	1.4E-3	1.4E-3	<b>1.3E-3</b>
Elevators	3.9E-3	3.2E-3	2.5E-3	2.4E-3	<b>2.3E-3</b>	<b>2.3E-3</b>	2.4E-3
Fried	1.920	1.496	1.554	1.396	1.067	1.155	<b>1.062</b>
House 16H	3.9E4	<b>3.4E4</b>	3.6E4	3.5E4	3.9E4	<b>3.4E4</b>	<b>3.4E4</b>
House 8L	3.5E4	3.1E4	3.2E4	3.1E4	3.3E4	3.1E4	<b>3.0E4</b>
Kinematics	0.198	0.165	0.176	0.158	<b>0.141</b>	0.149	<b>0.141</b>
Pole Telecomm	8.965	<b>6.377</b>	6.870	6.630	7.080	7.354	6.395
Puma32H	0.009	0.008	0.008	<b>0.007</b>	<b>0.007</b>	<b>0.007</b>	<b>0.007</b>
Puma8NH	3.424	3.266	3.216	3.210	3.202	3.199	<b>3.185</b>
Stock	1.41	1.02	1.08	0.993	0.885	0.782	<b>0.779</b>
Wisconsin Cancer	35.88	32.71	33.80	32.12	30.33	29.94	<b>29.90</b>
Average rank	5.05 (7)	3.32 (5)	3.68 (6)	2.53 (3)	2.58 (4)	1.84 (2)	<b>1.21 (1)</b>

Bold values indicate the performance of the top-ranked algorithm.

datasets. Furthermore, Friedman and Iman-Davenport statistical tests along with the Shaffer [34] test were used to verify if a configuration outperforms the others on most datasets. These statistical tests calculate the ranks achieved by each compared algorithm on the datasets. The null hypothesis  $H_0$  considers that all the selected algorithms are equal regarding their average ranks.  $H_0$  is accepted when the returned value, i.e., p-value, is less than a fixed significance level  $\alpha$ . The Shaffer post-hoc statistical test is adopted to find out which algorithm outperforms the others in the case where the  $H_0$  is rejected. The tested values of the considered parameters in the tuning procedure are: 50, 100 and 200 for the population size, and 8, 12, and 16 for the tree depth. Consequently, the parameter values which obtained the best average ranks on the test datasets and validated by Friedman, Iman-Davenport and Shaffer tests are selected, i.e., a population size of 50 for the upper-level, 100 for the lower-level, and a tree depth of 8, as given in Table III. We note that the remaining parameters include the number of generations, the crossover and mutation rates, and the elitism size. The compared algorithms were tested based on the best configurations given in their original papers which are also listed in Table III and can be found in the KEEL software [35]. In our work, all presented results, corresponding to an average of 31 runs, were obtained by 10-fold cross-validation.

## V. RESULTS AND DISCUSSION

### A. Comparison of the peer algorithms using the RMSE metric

Table IV presents the average RMSE values obtained by each of the studied algorithms on the 19 datasets over 31 runs. The last row of the table, named Average rank, gives the average rank for each algorithm based on RMSE values over all datasets. The obtained results show that BEMTI is the most stable algorithm, without instantaneous failures compared to all other algorithms. In fact, our proposed algorithm succeeded to attain the best average RMSE values, i.e., the lowest prediction error, in almost all cases of the tested datasets with 15 times of first rank and 4 times of second rank, computing



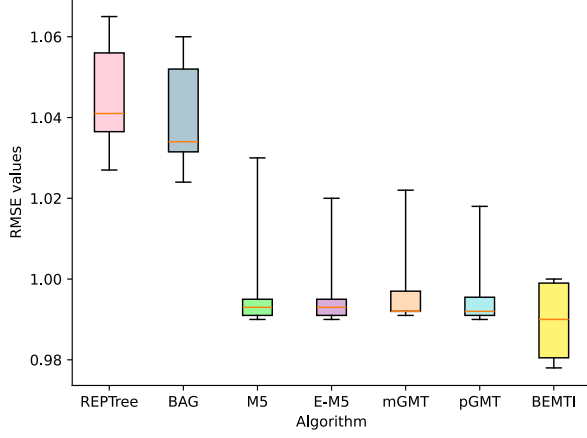


Fig. 4. RMSE values obtained by the compared algorithms over 31 runs using the 2D planes dataset.

the lowest average rank which equals to 1.21. The pGMT algorithm performs well by achieving the second best average rank, which equals to 1.84, with 8 times of first rank, 9 times of second rank and 1 time of third and sixth ranks. The third average rank of 2.53 was reached by E-M5 with 3 times of first rank, 8 times of second rank, 3 times of third rank and 5 times of other ranks followed by mGMT with a slightly greater average rank equal to 2.58 with 5 times of first rank, 3 times of second rank, 8 times of third rank and 3 times of other ranks. In Fig. 4, we illustrate the performance variation of the RMSE values obtained by REPTree, BAG, M5, E-M5, mGMT, pGMT and BEMTI using the 2D planes dataset. The horizontal axis corresponds to the algorithm and the vertical axis corresponds to the associated RMSE values over 31 runs. This comparison confirms the effectiveness of our BEMTI as opposed to the other algorithms of the experimental study. In fact, the distribution of the RMSE values shows that our BEMTI approach reaches the lowest RMSE median value which is located at 0.99 as opposed to the other compared algorithms, i.e., REPTree, BAG, M5, E-M5, mGMT and pGMT reaching median values respectively located at 1.041, 1.034, 0.993, 0.993, 0.992, and 0.992. We also note that the results obtained by the EAs considered in the experimental study could not compete with our proposed BEMTI.

Therefore, the efficiency of our BEMTI algorithm compared to the studied GMT-based algorithms (i.e., pGMT and mGMT) can be explained by the bi-level scheme adopted in our BEMTI. In fact, the upper-level search designs MTs where LM parameters, which are weights and biases, are defined by genetic variation operators. On the other side, the lower-level optimizes each of the LMs' parameters, and then returns the best LM having the optimal parameters' values to the upper-level in order to precisely and fairly compute its fitness, which is not the case of pGMT and mGMT. The latter can be seen as a single-level version of our BEMTI since they calculate

TABLE V  
RESULTS OF THE FRIEDMAN AND IMAN-DAVENPORT TESTS IN TERMS OF RMSE ( $\alpha = 0.05$ ).

Test	Parameters			
	Crit. value	Value	H0	p-value
Friedman	12.5916	<b>77.7636</b>	Rejected	<0.0001
Iman-Davenport	6.3125	<b>27.2261</b>	Rejected	<0.0001

TABLE VI  
THE AVERAGE MAE VALUES OBTAINED BY EACH OF THE CONSIDERED ALGORITHMS OVER 31 RUNS FOR EACH DATASET.

Dataset	REPTree	BAG	M5	E-M5	mGMT	pGMT	BEMTI
2D planes	<b>0.700</b>	0.743	0.732	0.731	0.732	0.731	0.729
Ailerons	1.3E-4	1.2E-4	<b>1.1E-4</b>	<b>1.1E-4</b>	<b>1.1E-4</b>	<b>1.1E-4</b>	<b>1.1E-4</b>
Bank32NH	0.079	0.073	0.069	0.069	0.070	<b>0.068</b>	0.069
Bank8FM	0.031	0.028	0.023	<b>0.022</b>	<b>0.022</b>	<b>0.022</b>	<b>0.022</b>
California Housing	7.8E4	6.3E4	9.1E4	6.2E4	5.9E4	5.9E4	<b>5.7E4</b>
CompAct	2.042	1.732	1.646	1.645	1.645	1.643	<b>1.638</b>
CompAct(s)	2.958	<b>2.415</b>	3.463	2.416	2.447	<b>2.415</b>	2.416
Delta Ailerons	1.4E-4	<b>1.3E-4</b>	<b>1.3E-4</b>	<b>1.3E-4</b>	<b>1.3E-4</b>	<b>1.3E-4</b>	<b>1.3E-4</b>
Delta Elevators	<b>1.0E-3</b>	1.1E-3	1.1E-3	1.1E-3	1.1E-3	1.1E-3	<b>1.0E-3</b>
Elevators	3.3E-3	2.7E-3	2.1E-3	<b>2.0E-3</b>	<b>2.0E-3</b>	<b>2.0E-3</b>	<b>2.0E-3</b>
Fried	1.411	1.099	1.421	1.026	0.784	0.849	<b>0.780</b>
House 16H	2.7E4	<b>2.3E4</b>	2.5E4	2.4E4	2.7E4	<b>2.3E4</b>	<b>2.3E4</b>
House 8L	2.5E4	3.1E4	2.3E4	2.3E4	2.4E4	2.3E4	<b>2.2E4</b>
Kinematics	0.162	0.135	0.144	0.129	<b>0.115</b>	0.122	<b>0.115</b>
Pole Telecomm	5.379	<b>3.826</b>	4.122	3.978	4.248	4.412	3.837
Puma32H	0.007	0.006	0.006	<b>0.005</b>	<b>0.005</b>	<b>0.005</b>	<b>0.005</b>
Puma8NH	2.773	2.645	2.604	2.600	2.593	2.591	<b>2.579</b>
Stock	0.916	0.663	0.702	0.645	0.575	0.508	<b>0.506</b>
Wisconsin Cancer	25.83	23.55	24.33	23.12	21.83	21.55	<b>21.52</b>
Avg Rank	4.53 (7)	3.37 (5)	3.58 (6)	2.42 (3)	2.47 (4)	1.84 (2)	<b>1.21 (1)</b>

Bold values indicate the performance of the top-ranked algorithm.

directly the fitness of each individual MT without optimizing its LMs' parameters. In order to verify the statistical validity of the obtained RMSE results, a statistical study is carried out on the selected algorithms. Firstly, we checked if there are significant differences between the selected algorithms based on the Friedman and Iman-Davenport test. As we can see in Table V, the Friedman and Iman-Davenport tests confirmed the existence of statistically significant differences since their generated values are higher than the critical value, which allows to reject the H0 with a p-value < 0,0001 for the Friedman test and a p-value < 0,0001 for the Iman-Davenport test. Consequently, we proceed with the Shaffer test which performs a comparison between all the selected algorithms. Results showed that BEMTI has the best behavior compared to all the remaining algorithms and that statistical differences exist between each selected algorithm and the other algorithms. Hence, we can conclude that our BEMTI is the best performing algorithm followed by pGMT, E-M5 and mGMT when solving the RT design problem, which is compatible with the obtained average rank values of the all selected algorithms.

#### B. Comparison of the peer algorithms using the MAE metric

The average MAE values over 31 runs, obtained by each of the considered algorithms using the 19 datasets are shown in Table VI. The last row of the table, i.e., Average rank, presents the average rank of each algorithm. The computed MAE values confirm the results of the obtained RMSE values (cf. section V-A) and generated similar results to those of RMSE since BEMTI maintains the best performance over

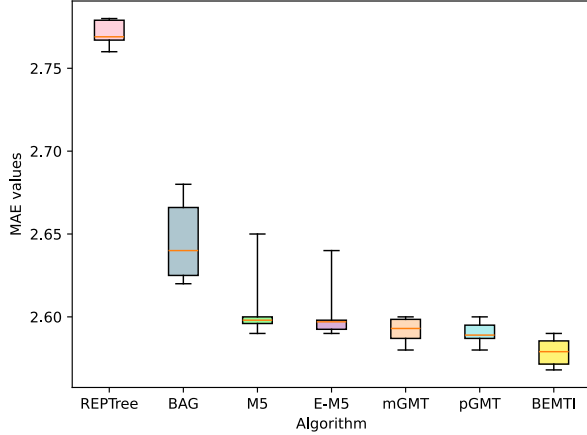


Fig. 5. MAE values obtained by the compared algorithms over 31 runs using the Puma8NH dataset.

all the compared algorithms with an average rank equal to 1.21 followed by pGMT, E-M5, and mGMT which achieved respectively average ranks equal to 1.84, 2.42 and 2.47, despite the difference between the two metrics since RMSE is more sensitive to outliers. Fig. 5 illustrates the MAE values obtained by the selected algorithms of the experimental study over 31 runs using the Puma8NH dataset. BEMTI outperforms all the other algorithms and attained the lowest median MAE value which is equal to 2.579, followed by pGMT (2.589), mGMT (2.593), E-M5 (2.597), M5 (2.598), BAG (2.64), and REPTree (2.769). The statistical validity of the obtained MAE results is checked by verifying the existence of significant differences between the compared algorithms using the Friedman and Iman-Davenport test as shown in Table VII. These first statistical tests validate the statistically differences by generating values which surpass the critical values allowing to reject the  $H_0$  ( $p\text{-value} < 0.0001$  for the Friedman test and  $p\text{-value} < 0.0001$  for the Iman-Davenport test). Then, we applied the Shaffer's test and the obtained results showed that our BEMTI performed well by presenting the best behavior in comparison to the selected algorithms and that there is no significant statistical differences with respect to the other algorithms. The MAE study validates the conclusion obtained by the RMSE study showing that BEMTI outperforms the compared algorithms from the greedy category, i.e., REPTree, BAG, M5 and E-M5 and from the EA category which are mGMT and pGMT, thus, confirming the added value of the use of a bi-level optimization to induce MTs along with the EA. These results can be explained by the fact that the lower-level of our BEMTI ensures a second optimization of the MT candidates by a further exploration of the search space to find the best sequence of LMs' coefficients of each upper-level solution as opposed to pGMT and mGMT, in which the EA is restricted to a single evaluation of the MTs and do not consider the optimization of the coefficients of the LMs. We can conclude that our proposed BEMTI is able to obtain a

TABLE VII  
ALGORITHMS COMPARISON MAE - FRIEDMAN AND IMAN-DAVENPORT TESTS ( $\alpha = 0.05$ ).

Test	Parameters			
	Crit. value	Value	H0	p-value
Friedman	12.5916	<b>60.1343</b>	Rejected	$< 0.0001$
Iman-Davenport	5.3678	<b>15.6163</b>	Rejected	$< 0.0001$

meaningful performance improvement compared to the state-of-the-art EAs.

## VI. CONCLUSION AND PERSPECTIVES

In this work, we proposed a new global approach called BEMTI to induce MTs as a solution to the regression problem based on an EA. Our algorithm relies on a bi-level architecture where each level intends to optimize a well-defined objective. Indeed, the upper-level evolves a set of MTs characterized by univariate or oblique tests at internal nodes and LMs in the leaves sought by an EA. In the lower-level, a population of LMs is evolved in order to find the best model which presents the optimal solution. The BEMTI method has been compared to four top-down RT induction algorithms as well as two EAs by computing the RMSE and MAE values applied on nineteen regression datasets. Results show the outperformance of our proposed approach and confirm the added value of the bi-level optimization along with EAs. Some threats to validity can be related to our approach which can impact the main findings of our work. Three types of threats to validity can be identified: (1) construct validity, (2) internal validity, and (3) external validity. Construct validity concerns the relationship between the theoretical aspect of the study and the observed results. In our work, we used two metrics to conduct the comparative study which are the RMSE and the MAE, since the former is sensitive to the outliers whereas the latter reports the errors equally. However, other metrics can be used for regression such as the  $R^2$  metric [2] which is able to indicate how well the predictions of the model can fit the data and, therefore, it can be a good measure. Internal validity guarantees that the obtained results represent the truth and are not affected by other hypotheses. The stochastic nature of our proposed BEMTI may present an internal threat to validity. The Shaffer test was conducted to mitigate this threat. However, a control method can be used in order to tune the parameters of our approach. External validity concerns the capability of the approach to be generalized to a broader context. Our BEMTI achieved competitive results compared to the other algorithms in our experimental study. Nevertheless, using other datasets from various domains would be more interesting. Following this work, several directions for future research are worth investigating. First, so far, BEMTI is designed to deal with single-output regression. Recent research suggests that multi-output learning is possible for numerical forecasting, especially with the considerable development of multi-tasking learning [36]. Second, another direction is domain adaptation for transfer learning. As BEMTI could be trained on a particular dataset



and provide desired outcomes, the change of the dataset would alter its performance. To transfer the ability of the learnt MT to make predictions on unseen but relatively similar datasets, domain adaptation [37] could be a wise choice.

## REFERENCES

- [1] H. Al-Sahaf, Y. Bi, Q. Chen, A. Lensen, Y. Mei, Y. Sun, B. Tran, B. Xue and M. Zhang, "A survey on evolutionary machine learning", *Journal of the Royal Society of New Zealand*, vol. 49.2, 205–228, May 2019.
- [2] Y. Tai, "A Survey Of Regression Algorithms And Connections With Deep Learning", *ArXiv*, vol. abs/2104.12647, April 2021.
- [3] M. Czajkowski and M. Kretowski, "A multi-objective evolutionary approach to Pareto-optimal model trees", *Soft Computing* vol. 23.5, pp. 1423–1437, March 2019.
- [4] L. Breiman, J. Friedman, R. Olshen, and C. Stone, "Classification and Regression Trees", Chapman and Hall/CRC, 1984.
- [5] P. Arora, H. Malik, and R. Sharma, "Wind speed forecasting model for northern-western region of India using decision tree and multilayer perceptron neural network approach", *Interdisciplinary Environmental Review*, vol. 19.1, pp. 13, January 2018.
- [6] J. R. Quinlan, "Learning with Continuous Classes", *Proceedings of Australian Joint Conference on Artificial Intelligence*, pp. 343–348, November 1992.
- [7] T. M. Sattarri, M. Pal, R. Mirabbasi, and J. Abraham, "Ensemble of M5 model tree-based modelling of sodium adsorption ratio", *Journal of AI and Data Mining*, vol. 6.1, pp. 68–78, March 2018.
- [8] M. Czajkowski and M. Kretowski, "The role of decision tree representation in regression problems – An evolutionary perspective", *Applied Soft Computing* vol. 48, pp. 458–475, November 2016.
- [9] G. Fan and J.B. Gray, "Regression Tree Analysis Using TARGET", *Journal of Computational and Graphical Statistics*, vol. 14.1, pp. 206–218, March 2005.
- [10] M. Czajkowski and M. Kretowski, "An evolutionary algorithm for global induction of regression trees with multivariate linear models", *International Symposium on Methodologies for Intelligent Systems*, vol. 6804, pp. 230–239, June 2011.
- [11] M. Czajkowski and M. Kretowski, "Global Induction of Oblique Model Trees: An Evolutionary Approach", *Artificial Intelligence and Soft Computing*, vol. 7895, pp. 1–11, June 2013.
- [12] M. Geerts, S. vanden Broucke and J. Weerdt, "An Evolutionary Geospatial Regression Tree", *Second International Workshop on Spatio-Temporal Reasoning and Learning*, September 2023.
- [13] T. Grubinger, A. Zeileis, and K-P. Pfeiffer, "evtree: Evolutionary Learning of Globally Optimal Classification and Regression Trees in R", *Journal of Statistical Software*, vol. 61.1, pp. 1–29, October 2014.
- [14] M. Elarbi, S. Bechikh, L. Ben Said and R. Datta, "Multi-objective Optimization: Classical and Evolutionary Approaches". In: S. Bechikh, R. Datta, A. Gupta (eds) *Recent Advances in Evolutionary Multi-objective Optimization. Adaptation, Learning, and Optimization*, vol. 20, Springer, Cham, 2017.
- [15] F. Y. Wu and K. K. Yen, "Applications of neural network in regression analysis", *Computers and Industrial Engineering*, vol. 23.1-4, pp. 93-95, November 1992.
- [16] S. Jun, "Evolutionary Algorithm for Improving Decision Tree with Global Discretization in Manufacturing", *Sensors*, vol. 21.8, pp. 2849, April 2021.
- [17] H. Drucker, C. J. C. Burges, L. Kaufman, and A. J. Smola, "Support vector regression machines", *Advances in Neural Information Processing Systems*, vol. 28.7, pp. 779–784, January 1997.
- [18] E. Frank, L. Trigg, G. Holmes, and I. H. Witten, "Naive Bayes for Regression", *Machine Learning*, vol. 41, pp 5–25, October 2000.
- [19] Y. Wang, and I. Witten, "Induction of model trees for predicting continuous classes", *Computer Science Working Paper Series*, vol. 96.23, October 1996.
- [20] D. Malerba, F. Esposito, M. Ceci, and A. Appice, "Top-down induction of model trees with regression and splitting nodes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26.5, pp. 612–25, June 2004.
- [21] A. Liaw and M. Wiener, "Classification and Regression by RandomForest", *R news*, vol. 2.3, pp. 18–22, December 2002.
- [22] L. Breiman, "Bagging predictors", *Machine Learning* vol. 24, pp. 123–140, August 1996.
- [23] R. E. Schapire, Y. Freund, P. Bartlett and W. S. Lee, "Boosting the margin: A new explanation for the effectiveness of voting methods", *Annals of Statistics*, vol. 26.5, pp. 1651–1686, October 1998.
- [24] L. Breiman, "Random forests", *Machine Learning*, vol. 45.1, pp. 5–32, January 2001.
- [25] B. Wang, H. K. Singh, and T. Ray, "An Evaluation of Simple Solution Transfer Strategies for Bilevel Multiobjective Optimization.", *2023 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, July 2023.
- [26] M. Czajkowski and M. Kretowski, "Evolutionary induction of global model trees with specialized operators and memetic extensions", *Information Sciences*, vol. 288, pp. 153–173, December 2014.
- [27] S. Chowdhury, Y. Linb, B. Liawc and L. Kerby, "Evaluation of Tree Based Regression over Multiple Linear Regression for Non-normally Distributed Data in Battery Performance", *ArXiv*, vol. arXiv:2111.02513, November 2021.
- [28] A. F. Brindle, "Genetic algorithms for function optimization", PhD dissertation, University of Alberta, 1981.
- [29] J. R. Koza, "Genetic programming: on the programming of computers by means of natural selection", *Statistics and Computing*, vol. 4, pp. 87–112, 1994.
- [30] K. Deb, "GeneAS: A robust optimal design technique for mechanical component design", *Evolutionary algorithms in engineering applications*, pp. 497–514, 1997.
- [31] L. J. Gleser, R. J. Carroll and P. P. Gallo, "The limiting distribution of least squares in an errors-in- variables regression model", *The Annals of Statistics*, vol. 15.1, pp. 220–233, March 1987.
- [32] K. Deb and A. Kumar, "Real-coded Genetic Algorithms with Simulated Binary Crossover: Studies on Multimodal and Multiobjective Problems", *Complex Systems*, vol. 9, pp. 431–454, 1995.
- [33] M. Juez-Gil, I. Nikolaevich Erdakov, A. Bustillo and D. Yurievich Pimenov, "A regression-tree multilayer-perceptron hybrid strategy for the prediction of ore crushing-plate lifetimes", *Journal of Advanced Research*, vol.18, pp. 173–184, July 2019.
- [34] J. P. Shaffer, "Modified Sequentially Rejective Multiple Test Procedures", *Journal of the American Statistical Association*, vol. 81.395, pp. 826–831, September 1986.
- [35] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez and F. Herrera, "KEEL Data-Mining Software Tool: Data Set Repository", *Integration of Algorithms and Experimental Analysis Framework, Journal of Multiple-valued Logic and Soft Computing*, vol. 17.2, pp. 255–287, January 2010.
- [36] D. Xu, Y. Shi, I. W. Tsang, Y-S. Ong, C. Gong and X. Shen, "A survey on multi-output learning", *IEEE transactions on neural networks and learning systems*, vol. 31.7, pp. 2409–2429, July 2020.
- [37] Z. Chen, D. Li, J. Liu and K. Gao, "Application of Gaussian processes and transfer learning to prediction and analysis of polymer properties", *Computational Materials Science*, vol. 216.17, pp. 111859, January 2023.