2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

A weighted ensemble of regression methods for gross error identification problem

Daniel Dobos[1,2], Truong Dang[1],
Tien Thanh Nguyen[1], John McCall[1]
[1]*National Subsea Centre, Robert Gordon University, Aberdeen, UK*
[2]*Luxembourg Center for Logistics and Supply Chain Management*
University of Luxembourg, Luxembourg
Email: d.dobos1@rgu.ac.uk

Allan Wilson[3], Helen Corbett[3],
Phil Stockton[3]
[3]*Accord ESL*
Aberdeen, UK

***Abstract*—In this study, we proposed a new ensemble method to predict the magnitude of gross errors (GEs) on measurement data obtained from the hydrocarbon and stream processing industries. Our proposed model consists of an ensemble of regressors (EoR) obtained by training different regression algorithms on the training data of measurements and their associated GEs. The predictions of the regressors are aggregated using a weighted combining method to obtain the final GE magnitude prediction. In order to search for optimal weights for combining, we modelled the search problem as an optimisation problem by minimising the difference between GE predictions and corresponding ground truths. We used Genetic Algorithm (GA) to search for the optimal weights associated with each regressor. The experiments were conducted on synthetic measurement data generated from 4 popular systems from the literature. We first conducted experiments in comparing the performances of the proposed ensemble using GA and Particle Swarm Optimisation (PSO), nature-based optimisation algorithms to search for combining weights to show the better performance of the proposed ensemble with GA. We then compared the performance of the proposed ensemble to those of two well-known weighted ensemble methods (Least Square and BEM) and two ensemble methods for regression problems (Random Forest and Gradient Boosting). The experimental results showed that although the proposed ensemble took higher computational time for the training process than those benchmark algorithms, it performed better than them on all experimental datasets.**

***Index Terms*—Gross error, ensemble method, regression, Genetic Algorithm, weighted ensemble**

I. Introduction

In the chemical industry, there is a need for high precision and error-free measurements to ensure the integrity of the operations. Reducing measurement errors and early fault detection can increase operational efficiency and save maintenance costs. Data reconciliation is a widely-used method in chemical processes to eliminate random noise from measurements in such a way that the system satisfies different physical constraints such as mass or energy balance. The reconciliation algorithms work under the hypothesis that the measurements contain only random errors. Gross errors (GEs) meanwhile are systematic errors and these reduce the effectiveness of reconciliation algorithms by biasing results away from true values [1].

To check the measurement integrity, many operators in the hydrocarbon industry apply simple rules to filter out corrupted measurements, such as checking for stuck meters, using simple thresholds, or using GE detection and identification methods. These methods are based on the statistical likelihood of the measurement containing an error using conversational laws, such as mass balance equations. It is noted that these tests can only be as effective as the mathematical models that they are based on, and they require prior knowledge, or an estimation of the distribution of statistical noise associated with each meter. Statistical-based tests such as Global Test (GT) were introduced in the early 1960s, and typically do not employ historical information of measurement data [2], [3]. With digitalisation, storing and logging data made much easier, it is important that the tests' performance can be improved by using a data-driven approach.

Machine learning (ML) is a popular topic with numerous applications in many industries, such as software engineering, medical imaging, insurance, and chemical engineering [3], [4]. ML algorithms find natural patterns in data that generate insight to make better decisions and predictions. One popular approach to improve the accuracy of ML models is ensemble learning. Ensemble models combine multiple learning algorithms, and the goal is to achieve a better prediction from the ensemble than the individual models. Ensembles can be described as homogeneous or heterogeneous. Homogeneous ensemble is a collection of classifiers or regressors of the same type, trained upon a different subset of the data. Heterogeneous ensemble is a set of different types of classifiers or regressors, trained upon the same dataset, and it is generally agreed that the more diverse the models are, the better a heterogeneous ensemble performs.

In ensemble learning, a combining method is used to combine the outputs of ensemble members to obtain a better prediction than that of individual members. For the regression problem, the combining method can be constructed by using a linear combination, with the weights assigned to the outputs of the individual models. While there are published papers using machine learning for the GEI problem with promising results [3], in this paper, we will introduce a novel weighted ensemble of regressors for same problem. The details and main contributions of our proposed method are as follows:

- We propose a heterogeneous ensemble of regressors

(EoR) to predict GEs in the measurement data in which different regression algorithms are trained on measurement datasets to generate the EoR. We also use the Stacking algorithm [5] to generate predictions for all training measurement instances and use these predictions to train the combining algorithm.
- The outputs of EoR are combined using a weighted combining method. The weights are found by minimising the differences between the predictions and the ground truth of the GEs on the training measurement instances. We use Genetic Algorithm (GA), a popular evolutionary computation method inspired by natural processes, to search for the optimal combining weights for each regressor.
- Experiments are conducted on 4 synthetic datasets generated from 4 popular systems in the literature [6]. The proposed method is compared with some popular benchmarks. The results indicate that our proposed method achieves the best results compared to the benchmark algorithms.
- We contributed an application of our ensemble regression method to real-world scenarios in the field of predictive modeling. By applying our novel weighted ensemble of regressors to practical problems, we demonstrated its effectiveness in improving predictions and highlighted its potential in the scope of the GEI problem.

II. Background and Related Work

A. Gross Error and Identification Techniques

In the past, numerous papers have discussed data reconciliation, which is a well-established mathematical and statistical technique used in allocation systems. It aims to derive the most accurate estimates of true measurement values while considering the preservation of physical conservation laws. These estimates account for minor imbalances in quantities such as mass and energy that may arise from random errors in measurements. When GE is present in the measurements, either due to instrument malfunction or process leaks, it can completely invalidate the statistical basis of data reconciliation techniques. Therefore, it is necessary to detect and identify any data which contains GE before applying a reconciliation method. There are four notable requirements when designing any GE detection and identification method [1] (i) detect the presence of one or more GE (the detection problem) (ii) identify and locate the single GE (the identification problem) (iii) identify and locate multiple gross errors (MGE) present in the system (the MGE identification problem) (iv) estimate the GE magnitude. In this study, we will focus on the fourth requirement.

To eliminate GE, we use several well-established tests in the literature. These methods mostly utilise hypothesis testing, and they are based on the hypothesis that the measurements only contain random errors. Test statistics are compared to a critical value, which is obtained from a significance level to determine whether the null hypothesis is rejected or not. The most widely used statistical test is the Global Test (GT) [2], where the result is based on the constraint residual vector and the covariance matrix of the measurements. This method only outputs an indication if a GE is present and does not support any information on the location or magnitude of the GEs. The Nodal Test also referred to as Constraint Test (CT) [1], operates on the same principles as GT. However, it conducts hypothesis testing on each constraint by analysing the diagonal elements of the constraint residuals. CT not only provides a result indicating the presence of a GE but also offers an estimation of the location by identifying the specific constraint(s) where the error occurs. Crowe [7] proposed the Maximum Power Nodal Test to enhance the effectiveness of CT, resulting in an increased probability of detecting GEs. Measurement test [7] uses chi-square distribution and normal statistics to test for overall errors in the measurement, for each node imbalance which is fully measured. Generalised Likelihood Ratio (GLR)[8] test is based on the likelihood ratio statistical test and provides a framework to identify the type of GE and is capable of detecting "leaks" in a steady state condition, which is an advantage over the previously mentioned methods.

For data-driven-based approaches, Reddy and Mavrovouniotis [9] used a 3-layer Neural Network (NN) to estimate the value of each measurement and its associated residual error. If the sum of the squares of the residuals does not fall within the established confidence limits, this sample is highly likely to contain a GE. Gerber et al. [10] considered GED as a binary classification problem and compared the GE detection results of 3 classifiers namely decision tree, linear, and quadratic discriminative analysis on the data generated from a simple two-product splitter process. Nguyen et al. [11] used the Fisher combination method to combine p-values output of several statistical tests, providing better results than using each statistical test. These methods, however, did not put any attention to the magnitude estimation of the GEs.

B. Ensemble Methods and Weighted Ensemble

Ensemble learning typically refers to a machine learning approach that combines multiple learning models to make a collaborated prediction that ideally outperforms those of the individual models. Ensemble learning can be used to solve classification problems (for sorting data into a set of classes or categories based on their characteristics) or regression problems (for predicting one or multiple so-called dependent variables with continuous decision-making based on the independent variables). Since the techniques aim to solve different problems, ensemble methods for classification and regression problems have been developed somewhat independently [12]. An extensive survey by Mendes-Moreira et al. [12] showed that many ensembles directly imported from the classification failed to demonstrate their viability for the regression problem.

In this study, we focus on ensemble aggregation which defines a strategy to combine the outputs of multiple models to obtain the final prediction. For the regression problem, the aggregation can be described as a weighted combination of the individual predictions as the following: $\hat{\mathbf{f}}(\mathbf{x}) = \sum_{i=1}^{K} [w_i(\mathbf{x}) *$

$\hat{f}_i(\mathbf{x})]$ are the weight $w_i(\mathbf{x})$ functions, $\hat{f}_i(\mathbf{x})$ is the prediction of i^{th} model among K models $(i = 1, ..., K)$, and $\hat{\mathbf{f}}(\mathbf{x})$ is the final combination.

Li et al. [13] divided weighting functions into two categories namely constant weights and dynamic weights. In the first category, the weights are constant and independent from the input. The main constant weight functions can be described using approaches such as Least Square (LS) [14], [15], Basic Ensemble Method (BEM) [12], and Generalised Ensemble Method (GEM) [12]. In LS, the ensemble combination is formulated as multiple linear regression problems for the weights of each different classifier. BEM averages the predicted values of K regressors as the following target function: $\hat{f}_{BEM}(\mathbf{x}) = \sum_{i=1}^{K}[\frac{1}{K} * \hat{f}_i(\mathbf{x})]$. The estimator of the BEM was rewritten as $\hat{f}_{BEM}(\mathbf{x}) = f(\mathbf{x}) - \frac{1}{K}\sum_{i=1}^{K} m_i(\mathbf{x})$, where $m_i(\mathbf{x}) = f(\mathbf{x}) - f_i(\mathbf{x})$. It is assumed that $m_i(\mathbf{x})$ are mutually independent with zero mean. The estimator for GEM [12] meanwhile is given by $\hat{f}_{BEM}(\mathbf{x}) = \sum_{i=1}^{K}[w_i(\mathbf{x}) * \hat{f}_i(\mathbf{x})] = f(\mathbf{x}) + \sum_{i=1}^{K}[w_i(\mathbf{x}) * \hat{m}_i(\mathbf{x})]$ in which $\sum_{i=1}^{K} w_i = 1, w_i = \sum_{j=1}^{K} c_{ij}^{-1} / \sum_{l=1}^{K}[\sum_{p=1}^{K} C_{lp}^{-1}], C_{ij} = E(m_i(\mathbf{x}) * m_j(\mathbf{x}))$. It is noted that in GEM, the weights are proportional to the validation error while considering the correlation of the error of the regressors. The dynamic weights approach meanwhile was first originally presented for the classification problem by Puuronen et al. [16] and then was adapted to the regression problems by Ronney et al. [17]. This approach includes 3 techniques namely Dynamic Selection (DS), Dynamic Weighting, and Dynamic Weighting with Selection. DS chooses the model to use which has the lowest cumulative error for the nearest neighbour to the test instance. DW assigns the weights based on the base models' performances on the nearest neighbour. The final prediction is the sum of the base models' predictions weighted with normalised weight values. Dynamic weighting with selection (DWS) uses the same localised data but adds an extra pruning step and discards the predictors which can be found in the upper half of the error interval. Some examples of dynamic weights approach are fuzzy logic for combining the predictions [18]. Reinforced Learning-based method [19], and Neural Networks (NN) -based estimation [13].

III. Proposed Method

For the GED problem, when the GE information of measurement data is given, we can train a supervised ML algorithm on the training data. Let $\mathbf{D}$ be the training set of N observations $\{(\mathbf{X}_n, \mathbf{Y}_n)\}_{n=1}^{N}$ where $\mathbf{X}_n = (x_{n1}, x_{n2}, ..., x_{nL})$ is a measurement including L stream values in the training set and $\mathbf{Y}_n = (y_{n1}, y_{n2}, ..., y_{nL})$ is its corresponding GE ground truth. The relationship between $\mathbf{X}_n$ and $\mathbf{Y}_n$ can be described by an unknown function f i.e., $\mathbf{Y}_n = f(\mathbf{X}_n)$. Supervised ML algorithms aim to propose an approximation (also called hypothesis) g for the function f. On that basis, we apply g to predict the GEs of unseen samples. In this study, we propose an EoR-based framework to predict the magnitude of GEs in the measurement data. The outputs of EoR are combined to obtain the final predicted GE for any test data. We propose to use weighted combining approach in which each regressor contributes a different weight to the combining result. Two following questions arise from the proposed framework:

- How can we construct an EoR to predict the GE?
- How can we find the weights for the combining method?

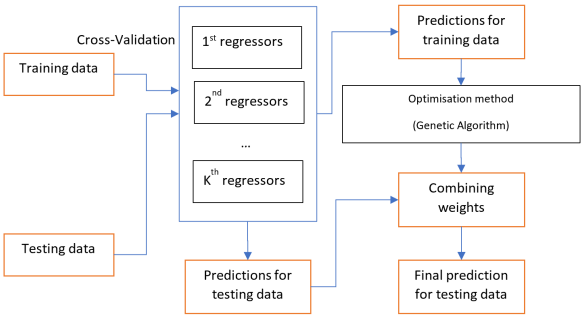

Fig. 1. Proposed ensemble of regressors for the GEI problem.

A. Ensemble of regressors

We proposed an EoR to solve the GEI problem. Let's denote $\mathbf{K} = \{\mathcal{K}_k\}_{k=1}^{K}$ as the set of K regression algorithms. In the ensemble, we train an EoR including K different regressors $\{g_k\}_{k=1}^{K}$ and then use a combining algorithm C on the EoR outputs to form the final decision-making: $g = C(\{g_k\}_{k=1}^{K})$. The ensemble of K regressors $\{g_k\}_{k=1}^{K}$ is generated by training K regression algorithms on the training set $\mathbf{D}$. The regression problem can be characterised by the following equation: $b_r = f_r(\mathbf{s}, \mathbf{\Theta}_r), b_r \in \mathbb{R}$ where $\mathbf{s}$ is the vector of the new observations (independent variables), b_r is the vector of the output(s) (dependent variables), and $f_r(.)$ is the regression function and $\mathbf{\Theta}_r$ are the function parameters.

In the GEI problem, $\mathbf{s} = (s_1, s_2, ..., s_L)$ is a vector of measurements of L stream and $b_r = (b_1, b_2, ..., b_L)$ is a vector of GEs in which b_l is associated with l^{th} stream. In this study, we aim to train a multi-output regressor $f_r(s_1, s_2, ..., s_L) = (b_1, b_2, ..., b_L)$ that maps a L vector of measurements to a L output vector of GEs. It is noted that there are several regression algorithms such as Random Forest and Linear Regression which natively support multi-outputs. On the other hand, not all regression algorithms support multioutput regression, one example is the support vector regression (SVR) which was developed based on the support vector machine (SVM) algorithm. To use regression models designed for predicting one value for multioutput regression, the multioutput regression problem is divided into multiple sub-problems. The most obvious way to do this is to split a multioutput regression problem into multiple single-output regression problems: $f_{r_1}(s_1, s_2, ..., s_L) = b_1, f_{r_2}(s_1, s_2, ..., s_L) = b_2, ..., f_{r_L}(s_1, s_2, ..., s_L) = b_L$. There are two main approaches to implementing this technique namely direct approach (developing a separate regression model for each output value to be predicted) and chain approach (predictions from the subsequent model were taken as part of the input to the next model). Although these approaches

can be used to train regressors in the proposed EoR, in this study, we use algorithms that support multi-outputs natively.

We then generate the predictions of each measurement instance in $\mathbf{D}$ by using the Stacking algorithm [20] . The training set is divided into T disjoint parts $\mathbf{D}_1, ..., \mathbf{D}_T, \mathbf{D}_i \cap \mathbf{D}_j = \emptyset, i \neq j$. For each $\mathbf{D}_t$ we denote its associated part as $\tilde{\mathbf{D}}_t = \mathbf{D} - \mathbf{D}_t$. The regression algorithm $\mathcal{K}_k$ trains on $\tilde{\mathbf{D}}_t$ to obtain a regressor $\tilde{g_{kt}} \cdot \tilde{g_{kt}}$ predicts for all instances in $\mathbf{D}_t$ to obtain the predicted GEs for instances in $\mathbf{D}_t$. This procedure runs through all parts $\tilde{\mathbf{D}}_t, t = 1, ..., T$ to obtain the predictions for all instances in $\mathbf{D}$. The predictions for all instances in the training set are given in an $N \times (K \times L)$ matrix (1)

$$\mathbf{B} = \begin{bmatrix} b_{11}^{(1)} & b_{12}^{(1)} & \dots & b_{1L}^{(1)} & \dots & b_{KL}^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{11}^{(n)} & b_{12}^{(n)} & \dots & b_{1L}^{(n)} & \dots & b_{KL}^{(n)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ b_{11}^{(N)} & b_{12}^{(N)} & \dots & b_{1L}^{(N)} & \dots & b_{KL}^{(N)} \end{bmatrix} \quad (1)$$

in which is $b_{ij}^{(n)}$ is the predicted GE for j^{th} stream of n^{th} instances of the training set, given by i^{th} regressors $(i = 1, \dots, K; j = 1, \dots, L; n = 1, \dots N)$

$$\mathbf{B}^{(n)} = \begin{pmatrix} b_{11}^{(n)} & b_{12}^{(n)} & \dots & b_{1L}^{(n)} & \dots & b_{KL}^{(n)} \end{pmatrix} \quad (2)$$

The next step is to train the combining algorithm on $\mathbf{B}$. In this study, we propose a weighted combining method in which each regressor contributes differently to the combined result by using different combining weights. The weights may vary for each regressor defined by $\mathbf{W} = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1L} & w_{K1} & w_{K2} \dots & w_{KL} \end{pmatrix}^T$ in which w_{ij} is the weight of the i^{th} regressor on the j^{th} stream. Different constraints can be imposed on w_{ij} such as Non-Negative Least Squares, i.e. $w_{ij} > 0$, Bounded Variable Least Squares $v_{ij} < w_{ij} < u_{ij}$, in which u_{ij} and are v_{ij} given upper and lower bounds of w_{ij}, and Bounded Variable with Constant Sum $-1 < w_{ij} < 1$, $\sum_{i=1}^{K} w_{ij} = 1$ [15]. In this study, the constraints for weights were set as $-1 < w_{ij} < 1$. Using the weights w, the predicted GE for j^{th} stream of an instance s is obtained by a linear combination of predictions b_{ij} for s and the associated weights as:

$$\hat{b}_j = \sum_{i=1}^{K} w_{ij} b_{ij} \quad (3)$$

in which b_{ij} is the predicted GE for j^{th} stream of instance s, given by i^{th} regressors $(i = 1, \dots, K; j = 1, \dots, L)$.

B. Data generation

Training data: In the model in Fig 1, the training dataset is fed to an EoR so that they can be trained to approximate the relationship between measurements and their associated GEs. In this study, we generated training data on 4 systems collected from the literature (Fig 2-5) [3][6]. Fig 2-5 show the structure of the experimental systems (parallel stream, recycled stream, or measurement). In these systems, the true values of measurements and the variances of random errors associated with all streams were given. First, random errors with a normal distribution (zero mean and given variance) for each stream were generated and added to the true measurement to create the base case data i.e., no-bias data. Then biases were generated for all streams and added to those base cases. In this work, each bias was generated under a uniform distribution between -25% to +25% of the associated measurement value. Uniform distribution was chosen, because to our best knowledge, there is no data available on the distribution of the gross error, and it could depend on the type of sensor, working conditions and many external factors [3]. The training data for a system of m streams contains 1000 samples with non-bias, 10 samples with GE on the m^{th} stream. Hence, the training data includes $m * 10 + 1000$ samples.

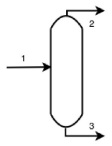

Fig. 2. Separator equipment used in GED of System 1 (S1)

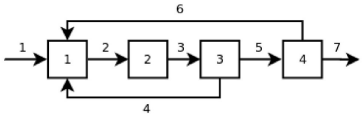

Fig. 3. Heat exchanger network with recycle flowsheet used in GED of System 2 (S2)

Testing process: In the model in Fig 1, a test sample is first passed through the data preparation process. In this study, test samples were also generated on 4 systems to study the performance of the proposed EoR. We generated one test dataset for each system in which each GE was generated under a uniform distribution between -5% and +5% of the associated measurement value (as small errors will be harder to detect). The data generation for testing data is the same as the training data generation. According to a system with m streams, we generated the test data of $m * 10 + 1000$ samples in which 1000 samples with no-GE, 1000 samples with GE on the m^{th} stream.

C. Optimisation approach

In this section, we propose an approach to obtain combining weight $\mathbf{W}$ for the ensemble framework. Since the ground truths of the GEs of training instances are known in advance, the weights of regressors should be introduced to minimise the difference between the predicted GE and the ground truth. Based on the predictions in (1) and weighted combining in (3) , we obtain the combined result for n^{th} training instance as:

$$\hat{b}_j^{(n)} = \sum_{i=1}^{K} w_{ij} b_{ij}^{(n)} \quad (4)$$

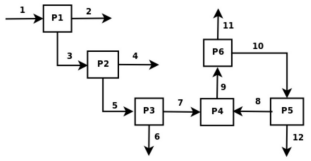

Fig. 4. Generic mass balance flowsheet used in GED of System 3 (S3)

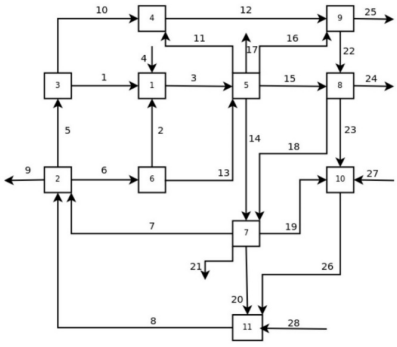

Fig. 5. Steam metering system flowsheet used in GED of System 4 (S4)

in which $\hat{b}_j^{(n)}$ is the predicted GE for j^{th} stream of n^{th} instances of the training set. The loss function as the total difference computed on all streams of all training instances is defined as:

$$\begin{aligned}\mathcal{L} &= \frac{\sum_{n=1}^{N}\sum_{j=1}^{L}(\hat{b}_j^{(n)} - y_{nj})^2}{N \times L} \\ &= \frac{\sum_{n=1}^{N}\sum_{j=1}^{L}(\sum_{i=1}^{K} w_{ij} b_{ij}^{(n)} - y_{nj})^2}{N \times L}\end{aligned} \tag{5}$$

in which $\mathcal{L}$ is the loss function and y_{nj} is the ground truth GE for the j^{th} stream of the n^{th} instance of the training set. The combining weight is obtained by minimising the loss function in (5). The proposed weighted combining model is different from other weighted combining models for regression problems. Compared to the LS method, its combining weights are obtained by solving the least square $min_{\mathbf{W}} \|\mathbf{BW} - \mathbf{Y}\|$ in which $\mathbf{W}$ is a $(KL \times L)$ matrix of combining weights and $\mathbf{Y}$ is a $(N \times L)$ matrix of the ground truth GE of all streams of all instances. The combining model in GEM meanwhile uses K weights computed on validation error while considering the correlation of the error of the regressors.

In this study, we used Genetic Algorithm to search for the optimal combining weights for the proposed ensemble system. The proposed GA in this paper consists of three operators: SELECTION, CROSSOVER, and MUTATION.

REPRESENTATION: We introduce a $L \times K$ vector to encode combining weights. It is noted that the constraints for weights were set as $-1 \leq w_{ij} \leq 1$ and the length of the representation depends on the number of regressors K and the number of streams L.

SELECTION: We utilize the roulette wheel selection procedure to balance fitness-based criteria and randomness [21].

Algorithm 1 Creating predictions for training data and EoRs

Require: Training data $\mathbf{D}$, number of cross-validation folds T, K regression algorithms $\{\mathcal{K}_k\}_{k=1}^{K}$
Output: The prediction $\mathbf{B}$ and the trained regressors $\{g_k\}_{k=1}^{K}$
1: Train K regressors $\{g_k\}_{k=1}^{K}$ on $\mathbf{D}$ using $\{\mathcal{K}_k\}_{k=1}^{K}$
2: $\mathbf{B} = \emptyset$
3: Divide $\mathbf{D}$ into T folds: $\mathbf{D} = \mathbf{D}_1 \cup \mathbf{D}_2 \cup \cdots \cup \mathbf{D}_T (\mathbf{D}_i \cap \mathbf{D}_j = \emptyset, i \neq j)$
4: **for** t from 1 to T **do**
5: $\tilde{\mathbf{D}}_t = \mathbf{D} - \mathbf{D}_t$
6: Train $\{\tilde{g}_{kt}\}_{k=1}^{K}$ on $\tilde{\mathbf{D}}_t$ using $\{\mathcal{K}_k\}_{k=1}^{K}$
7: Use $\{\tilde{g}_{kt}\}_{k=1}^{K}$ to predict $\mathbf{D}_t$
8: Add the results to $\mathbf{B}$
9: **end for**
10: **return** $\mathbf{B}$ and $\{g_k\}_{k=1}^{K}$

Algorithm 2 Fitness evaluation

Require: The candidate $\mathbf{W} = \{w_{ij}\}(1 \leq i \leq K, 1 \leq j \leq L)$ where K is the number of regressors and L is the number of streams, prediction $\mathbf{B}$, GE ground truth $\mathbf{Y}$.
Output: The fitness value of $\mathbf{W}$
1: $pred = \emptyset$
2: **for each** $\mathbf{B}^{(n)} = \left(b_{11}^{(n)} \quad b_{12}^{(n)} \quad \ldots b_{1L}^{(n)} \quad b_{KL}^{(n)}\right)$ in $\mathbf{B}$ **do**
3: **for** j from 1 to L **do**
4: $\hat{b}_j^{(n)} = \sum_{i=1}^{K} w_{ij} b_{ij}$
5: **end for**
6: Add $\hat{b}_j^{(n)}$ to $pred$
7: **end for**
8: Calculate $fitness$ by using (5) with $pred$ and $\mathbf{Y} = \{y_{nj}\}$
9: **return** $fitness$

At each generation, a number of parents are selected based on their fitness values to generate offspring. For parameters' settings, the population size is 100, and the maximum number of generations is set to 500 based on the experiments in [22].

CROSSOVER: We used uniform crossover with a crossover possibility P_c (was set to 0.9 in this study) to generate offspring from their parents. If a randomly generated crossover rate is smaller than P_c , the parent will be selected to generate offsprings.

MUTATION: A mutation probability parameter P_m (was set to 0.1 in this study) is used to direct the mutation process. Each element of the first offspring is replaced by a random number within the boundary range [-1, 1] with a probability P_m. Meanwhile, each element of the second offspring is replaced by a random number within the range of the corresponding elements in its parents with P_m.

Algorithm 1 describes the procedure for creating predictions for training data and EoR. In line 1, EoR $\{g_k\}_{k=1}^{K}$ are obtained by training $\{\mathcal{K}_k\}_{k=1}^{K}$ on $\mathbf{D}$. From line 3 to 9, we used the Stacking algorithm to generate predictions $\mathbf{B}$ for training data. In detail, $\mathbf{D}$ is divided into T disjoined parts. For each fold t, each regressor is trained on the remainder of $\mathbf{D}_t$ denoted by $\tilde{\mathbf{D}}_t$ then predict for instances in $\mathbf{D}_t$. The prediction is added to $\mathbf{B}$ to return with $\{g_i\}_{i=1}^{K}$.

Algorithm 2 describes the fitness evaluation procedure for a given candidate. From line 2 to 7, for prediction $\mathbf{B}^{(n)}$ for n^{th} instance in $\mathbf{B}$, we obtain the combining result using the weights $\mathbf{W}$ (lines 3-5). After obtaining the predicted GE for all training instances, the fitness value is calculated using equation

Algorithm 3 Training process with Genetic Algorithm to search for optimal combining weights

Require: Training data **D**, number of cross-validation folds T, K regression algorithms $\{\mathcal{K}_k\}_{k=1}^{K}$, the number of streams L, population size $popSize$, number of generations $maxGen$, crossover probability P_c, mutation probability P_m, elitist ratio E_r, parent selection percentage P_r.
Output: the optimal combining weights $\mathbf{W}_{opt}$
1: Use Algorithm 1 to create **B** and EoR $\{g_k\}_{k=1}^{K}$
2: Randomly initialize $popSize$ candidates.
3: $N_{elite} = |popSize * E_r|$, $N_{par} = |popSize * P_r|$
4: **for** gen from 1 to $maxGen$ **do**
5: **for** pop from 1 to $popSize$ **do**
6: Use Algorithm 2 to calculate the fitness of the pop^{th} candidate $\mathbf{W}$
7: **end for**
8: **for** pop from 1 to $popSize$ **do**
9: $\mathbf{prob}_{pop} = \frac{fitness_{pop}}{\sum_{j=1}^{popSize} fitness_j}$
10: **end for**
11: $Parent = \emptyset, EffParent = \emptyset$
12: Add N_{elite} candidate to $Parent$
13: Add $N_{par} - N_{elite}$ candidate to $Parent$ by using roulette wheel selection
14: **for** a candidate **in** $Parent$ **do**
15: **if** $Rand() \leq P_c$ **then**
16: Add candidate to $EffParent$
17: **end if**
18: **end for**
19: Add all candidates in $Parent$ to the new population
20: $j = N_{par} + 1$
21: **while** $j \leq popSize$ **do**
22: Choose two random candidates $\mathbf{W}^{(1)} = \begin{pmatrix} w_{11}^{(1)} & w_{12}^{(1)} & \dots & w_{L}^{(1)} & \dots & w_{KL}^{(1)} \end{pmatrix}$ and $\mathbf{W}^{(2)} = \begin{pmatrix} w_{11}^{(2)} & w_{12}^{(2)} & \dots & w_{L}^{(2)} & \dots & w_{KL}^{(2)} \end{pmatrix}$ in the $EffParent$ set
23: **for** k **from** 1 **to** K **do**
24: **for** l **from** 1 **to** L **do**
25: **if** $Rand() \leq 0.5$ **then**
26: Obtain $w_{kl}^{\prime(1)} = w_{kl}^{(2)}$ and $w_{kl}^{\prime(2)} = w_{kl}^{(1)}$
27: **end if**
28: **end for**
29: **end for**
30: **for** k **from** 1 **to** K **do**
31: **for** l **from** 1 **to** L **do**
32: **if** $Rand() \leq P_m$ **then**
33: Replace $w_{kl}^{\prime(1)}$ with a random number in the boundary range [-1,1]
34: **end if**
35: **end for**
36: **end for**
37: **for** k **from** 1 **to** K **do**
38: **for** l **from** 1 **to** L **do**
39: **if** $Rand() \leq P_m$ **then**
40: Replace $w_{kl}^{\prime(2)}$ with a random number in the range $[min(w_{lk}^{(1)}, w_{lk}^{(2)}), max(w_{lk}^{(1)}, w_{lk}^{(2)})]$
41: **end if**
42: **end for**
43: **end for**
44: Add $\mathbf{W}^{\prime(1)} = \{w_{kl}^{\prime(1)}\}$ and $\mathbf{W}^{\prime(2)} = \{w_{kl}^{\prime(2)}\}$ to the new population
45: $j = j + 2$
46: **end while**
47: **end for**
48: **return** $\mathbf{W}_{opt}$ **from** $maxGen$ generation

(5) (line 8).

Algorithm 3 describes the training process with GA to search for optimal combining weights. In line 1, Algorithm 1 is called to create the prediction B and the EoR $\{g_k\}_{k=1}^{K}$. From lines 2-3, the population is initialized, and the number of elites and parents are calculated. Afterward, for each generation, the fitness value of each candidate is first calculated using Algorithm 2 (lines 5-7), then the probabilities for roulette wheel selection are calculated (lines 8-10). In line 11, $Parent$ and $EffParent$ set are initialized in which $Parent$ denotes the set of candidates in the current generation which would be added to the next generation, while $EffParent$ denotes the set of candidates which will be used to generate offsprings. In lines 12 and 13, the parents are selected, with the first N_{elite} candidates are chosen from the best candidates, while the remaining are chosen using roulette wheel selection. From lines 14-18, the candidates in $Parent$ are added to $EffParent$ based on the crossover probability P_c, and in line 19, the candidates in $Parent$ are added to the new population. Then, from lines 20-46, the candidates are randomly selected from $EffParent$ set, and crossover and mutations are applied until a total of $popSize$ candidates have been obtained. In detail, two candidates $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ are randomly chosen from the $EffParent$ set, and their offspring, denoted as $W^{\prime(1)}$ and $W^{\prime(2)}$. Uniform crossover is performed from lines 23-29, in which the values of each dimension in $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ are swapped with a probability of 0.5. For mutation, firstly each element $w_{kl}^{\prime(1)}$ in the first offspring is replaced by a random number within the boundary range [-1, 1] (lines 30-36) with probability P_m . Meanwhile each element $w_{kl}^{\prime(2)}$ in the second offspring is replaced by a random number within the range of the corresponding elements in the parents with probability P_m (lines 37-43). Two offsprings $\mathbf{W}^{\prime(1)}$ and $\mathbf{W}^{\prime(2)}$ then are added to the new population. After $maxGen$ generations, the algorithm returns the optimal candidate $\mathbf{W}_{opt}$.

Algorithm 4 Evaluating test instance

Require: The best candidate $\mathbf{W}_{opt} = \{w_{ij}\}(1 \leq i \leq K, 1 \leq j \leq L)$, test instance $\mathbf{x}$, the EoR $\{g\}_{i=1}^{K}$
Output: The predicted GE for $\mathbf{x}$
1: $pred = \emptyset$
2: let $\begin{pmatrix} b_{11} & b_{12} & \dots & b_{1L} & \dots & b_{KL} \end{pmatrix}$ be the predictions of $\{g\}_{i=1}^{K}$ on $\mathbf{x}$
3: **for** j from 1 to L **do**
4: $\hat{b}_j = \sum_{i=1}^{K} w_{ij} b_{ij}$
5: **end for**
6: **return** $\{\hat{b}_j\}_{j=1}^{L}$

Algorithm 4 describes the evaluation process on the test set. The inputs of the algorithm consist of the best candidate $\mathbf{W}_{opt} = \{w_{ij}\}(1 \leq i \leq K, 1 \leq j \leq L)$, test instance $\mathbf{x}$, the trained regressors $\{g_k\}_{k=1}^{K}$.

IV. Experimental Studies

A. Experimental Settings

The proposed EoR was generated by training two regression algorithms namely Random Forest and Gradient Boosting. Random Forest was reported as one of the best methods among more than 150 ML methods in [23] while Gradient Boosting was reported as a reliable method for regression problems [24]. We used the Sklearn library with default parameters' values to implement these algorithms. To generate **B** in (1), a 5-fold Cross-Validation on the training measurement data was applied.

The performances of all experimental methods on the test samples were reported according to two performance metrics namely mean absolute error (MAE) and mean square error (MSE). We used the Friedman test to test the null hypothesis that all methods perform equally on the datasets. If the null hypothesis is rejected (that means the performances of all methods are different), we conducted the Nemenyi post-hoc

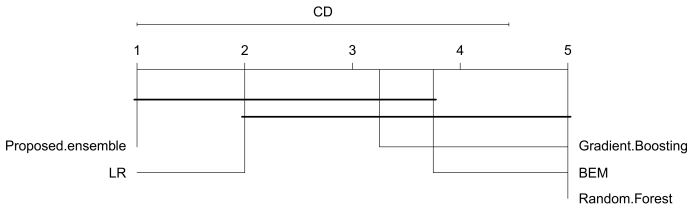

Fig. 6. The Nemenyi test result on MSE

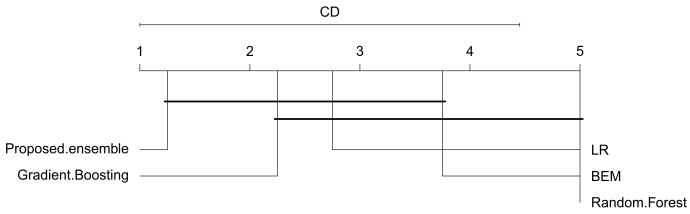

Fig. 7. The Nemenyi test result on MAE

TABLE I
THE MSE OF THE PROPOSED ENSEMBLE (2 REGRESSORS) AND BENCHMARK ALGORITHMS

Dataset	Random Forest	Gradient Boosting	BEM	LS	Proposed ensemble
S1	0.021055	0.019602	0.019701	0.019442	0.019417
S2	0.089433	0.073892	0.077728	0.055522	0.054523
S3	8.954313	6.209644	6.840754	3.918852	3.718571
S4	0.565045	0.522243	0.516814	0.507476	0.500162

TABLE II
THE MAE OF THE PROPOSED ENSEMBLE (2 REGRESSORS) AND BENCHMARK ALGORITHMS

Dataset	Random Forest	Gradient Boosting	BEM	LS	Proposed ensemble
S1	0.078704	0.075874	0.076407	0.076102	0.075869
S2	0.152985	0.128022	0.138561	0.112818	0.109807
S3	1.045511	0.732430	0.873343	0.699505	0.646974
S4	0.166957	0.132138	0.148205	0.158130	0.142902

test for all pairwise comparisons based on the rankings of experimental methods on all experimental test datasets. In this work, the confidence level of the Nemenyi test was set to 0.05.

B. Comparison with the Benchmark Algorithms

We compared the proposed ensemble to 4 benchmark algorithms: two ensemble members (Random Forest and Gradient Boosting), BEM, and LS method. The comparison of two ensemble members aims to show the advantage of EoR over single regressors. Meanwhile, the comparison to two weighted EoR namely BEM and LS aim to demonstrate the advantage of the proposed combining weight method in our method. Both BEM and Least Square were implemented with an ensemble of two members Random Forest and Gradient Boosting like the proposed method. We run each method one time and reported the results in Tables 1 and 2. Since the p-values of the Friedman test corresponding to MAE and MSE are all smaller than a given significant level, we then reject the null hypothesis that all methods performed equally. We run the Nemenyi post-hoc test to compare each pair of methods. The results of the Nemenyi test are shown in Fig 6 and 7.

For MSE, the Nemenyi test result in Fig 6 shows that the proposed ensemble ranks first among all experimental methods and is better than Random Forest, followed by LS and Gradient Boosting. The poorest method in our experiments was Random Forest (with rank value 5). In detail, the proposed ensemble ranks first on all datasets. On S1, S2, and S4 datasets, the MSE of the proposed ensemble was slightly better than the second-best method on those datasets (0.019417 vs. 0.019442 of LS on S1, 0.054523 vs. 0.055522 of LS on S2, and 0.500162 vs. 0.05075 of LS on S4). On S3 dataset, the proposed ensemble outperforms LS with more significance, 3.718 vs. 3.9189. Compared to BEM, the proposed ensemble yields significantly better results on S3 dataset (3.7186 vs. 6.8408). BEM simply computes the mean of predictions without attention to the difference in the performance of each regressor of the ensemble, thus this method obtained poorer results than LS and our method in the experiment.

For MAE, the Nemenyi post-hoc test result in Fig 7 shows that the proposed ensemble ranks first and is better Random Forest. The proposed method is worse than Gradient Boosting, the second-best method on S4 dataset (0.142902 vs. 0.132138) while outperforms that method on S2 and S3 dataset (0.109807 vs. 0.128022 on S2 and 0.646974 vs. 0.732430 on S3). The proposed ensemble performs slightly better than LS on S1 and S2 datasets while the MAE differences of those methods are significant on S3 dataset. Random Forest continues to be the poorest method in our experiments corresponding to MAE. The outstanding of the proposed ensemble can be explained by the weighted combining method.

C. Different Number of Regressors

We compared the proposed ensemble to the benchmark algorithms with a different number of regressors. In this experiment, we created ensemble systems with 3 regressors (Bayesian Ridge Regression algorithm, Ridge Regression algorithm, and Histogram-Based Gradient Boosting (HGB) for regression). Table 3 and 4 show the experimental results of all experimental methods. It is observed that the proposed ensemble once again is better than the 3 regressor members and two EoRs concerning both MSE and MAE. For MSE, the proposed ensemble ranked first on all datasets and for MAE a similar pattern was observed except on S1 dataset where the proposed ensemble's performance is slightly better than that of HGB (0.082068 vs. 0.078093). LS, another weighted combining method, was worse than the proposed method on all datasets. This method, even, was worse than the three regressor members on S4 dataset. These results indicated the advantage of our method with a different EoR.

We compared the training and testing time of the two weighted ensemble methods (the proposed method and LS). For the training process, LS and the proposed method took the same time in generating the prediction $\mathbf{B}$ in (1) and

TABLE III
THE MSE OF THE PROPOSED ENSEMBLE (3 REGRESSORS) AND BENCHMARK ALGORITHMS

Dataset	Bayesian Ridge	Ridge	HGB	BEM	LS	Proposed ensemble
S1	0.024818	0.024774	0.020179	0.020089	0.019770	0.019526
S2	0.057670	0.057846	0.108532	0.064989	0.081334	0.054844
S3	5.182465	5.197063	8.625569	5.002093	4.062159	3.877229
S4	0.933948	0.936405	0.681322	0.676290	2.504710	0.607045

TABLE IV
THE MSE OF THE PROPOSED ENSEMBLE (3 REGRESSORS) AND BENCHMARK ALGORITHMS

Dataset	Bayesian Ridge	Ridge	HGB	BEM	LS	Proposed ensemble
S1	0.117805	0.117670	0.078093	0.102706	0.083277	0.082068
S2	0.157104	0.157595	0.170890	0.158690	0.164433	0.123669
S3	1.425069	1.428178	0.968218	1.256697	0.896520	0.790358
S4	0.583918	0.584691	0.168820	0.442076	0.851757	0.282238

generating the EoR. However, the proposed ensemble took higher computational time in obtaining combining weights than the LS method because of using GA for searching. On S3 dataset, two methods took about 344 seconds in generating **B** and 88 seconds in generating the EoR; the proposed method took about 100 seconds in obtaining the optimal weights while LS took less than 1 second only. On the other hand, the proposed method took less testing time in the testing process compared to the LS method because of the small size of combining weights ($L \times K$ -matrix of the proposed method vs. $(LK) \times K$ -matrix of the LS method). On S3 dataset, the proposed method took 1.22 seconds to test all test instances while the LS method took 1.8 seconds on the same task.

V. CONCLUSIONS

In this paper, we proposed an EoR to predict GE magnitudes on measurement data. The ensemble includes a number of regressors obtained by training different regression algorithms on the training data including measurement instances and their associated GE ground truths. The outputs of EoR are combined by a weighted combining method showing how regressors contribute to the combining result. We modelled the weights search problem as an optimisation problem by minimising the differences between the predicted GE for instances in the training data and the corresponding GE ground truths. Three GA operators were proposed to search for the optimal combining weight of each regressor, including roulette wheel selection for breeding, single-point crossover to generate new offspring, and random mutation on each offspring.

We compared two optimisation methods GA and PSO with the proposed ensemble. The experimental results on 4 datasets generated from 4 popular systems in the literature showed that the proposed ensemble using GA performed better than that using PSO. Compared to two ensemble methods (Random Forest and Gradient Boosting) and two weighted combining ensemble methods (Least Square and BEM), the proposed ensemble outperforms those methods with different margins.

REFERENCES

[1] C. Jordache, S. Narasimhan, "Data reconciliation & gross error detection: an intelligent use of process data,", Gulf Professional Publishing, 1999.

[2] P. Reilly, R. Carpani, "Application of statistical theory of adjustment to material balances,", 13th Canadian Chem. Eng. Conference, 1963.

[3] D. Dobos et al., "A comparative study of anomaly detection methods for gross error detection problems," Computers & Chemical Engineering, vol. 175, p. 108263, 2023.

[4] T. Dang, A. V. Luong, A. W. C. Liew et al., "Ensemble of deep learning models with surrogate-based optimization for medical image segmentation," in 2022 IEEE Congress on Evolutionary Computation (CEC), 2022, pp. 1–8.

[5] T. T. Nguyen, M. T. Dang, V. A. Baghel et al., "Evolving interval-based representation for multiple classifier fusion," Knowledge-based systems, vol. 201, p. 106034, 2020.

[6] E. C. do Valle, R. de Araújo Kalid, A. R. Secchi et al., "Collection of benchmark test problems for data reconciliation and gross error detection and identification," Computers & Chemical Engineering, vol. 111, pp. 134–148, 2018.

[7] C. Crowe, Y. G. Campos, A. Hrymak, "Reconciliation of process flow rates by matrix projection. Part I: linear case," AIChE Journal, vol. 29, no. 6, pp. 881–888, 1983.

[8] S. Narasimhan, R. S. Mah, "Generalized likelihood ratio method for gross error identification," AIChE Journal, vol. 33, no. 9, pp. 1514–1521, 1987.

[9] V. Reddy, M. Mavrovouniotis, "An input-training neural network approach for gross error detection and sensor replacement," Chemical Engineering Research and Design, vol. 76, no. 4, pp. 478–489, 1998.

[10] E. F. Gerber, L. Auret, C. Aldrich, "The application of classification methods to the gross error detection problem," IFAC Proceedings Volumes, vol. 47, no. 3, pp. 11464–11469, 2014.

[11] T. T. Nguyen, J. McCall, A. Wilson et al., "Evolved ensemble of detectors for gross error detection," in GECCO, 2020, pp. 281–282.

[12] J. M.-Moreira, C. Soares, A. M. Jorge et al., "Ensemble approaches for regression: A survey, " ACM Comput. Surv., vol. 45, no. 1, pp. 1–10, Nov. 2012.

[13] K. Li, W. Liu, K. Zhao et al., "A novel dynamic weight neural network ensemble model," International Journal of Distributed Sensor Networks, vol. 11, no. 8, p. 862056, 2015.

[14] Y. Ren, L. Zhang, P. N. Suganthan, "Ensemble classification and regression-recent developments, applications and future directions," IEEE Computational intelligence magazine, vol. 11, no. 1, pp. 41–53, 2016.

[15] T. T. Nguyen, M. T. Dang, A. W. Liew et al., "A weighted multiple classifier framework based on random projection," Information Sciences, vol. 490, pp. 36–58, 2019.

[16] S. Puuronen, V. Terziyan, A. Tsymbal, "A dynamic integration algorithm for an ensemble of classifiers," in ISMIS, 1999, pp. 592–600.

[17] N. Rooney, D. Patterson, S. Anand et al., "Dynamic integration of regression models," in Multiple Classifier Systems: 5th International Workshop, 2004, pp. 164–173.

[18] A. Bagherinia, B. M. Bidgoli, M. Hosseinzadeh et al., "Reliability-based fuzzy clustering ensemble," Fuzzy Sets and Systems, vol. 413, pp. 1–28, 2021.

[19] S. K. Perepu, B. S. Balaji, H. K. Tanneru et al., "Dynamic Selection of Weights of Ensemble Models Using Reinforcement Learning for Time-Series Forecasting," in Advances in Information and Communication: Proceedings of the 2021 Future of Information and Communication Conference (FICC), Volume 2, 2021, pp. 613–624.

[20] K. M. Ting, I. H. Witten, "Issues in stacked generalization," Journal of artificial intelligence research, vol. 10, pp. 271–289, 1999.

[21] T. T. Nguyen et al., "Evolving an optimal decision template for combining classifiers," in Neural Information Processing: 26th International Conference, ICONIP 2019, pp. 608–620.

[22] T. Dang, T. T. Nguyen, J. McCall et al., "Ensemble Learning based on Classifier Prediction Confidence and Comprehensive Learning Particle Swarm Optimisation for Medical Image Segmentation," in IEEE Symposium Series on Computational Intelligence, 2022, pp. 269–276.

[23] M. Fernández-Delgado, E. Cernadas, S. Barro et al., "Do we need hundreds of classifiers to solve real-world classification problems?", J. Mach. Learn. Res., vol. 15, no. 1, pp. 3133–3181, Jan. 2014.

[24] H. Seto et al., "Gradient boosting decision tree becomes more reliable than logistic regression in predicting probability for diabetes with big data," Scientific Reports, vol. 12, no. 1, p. 15889, 2022.