Prediction of Material Removal Rate for Chemical Mechanical Planarization Using Decision Tree-Based Ensemble Learning

Zhixiong Li
Department of Mechanical and Aerospace Engineering,
University of Central Florida,
Orlando, FL 32816
e-mail: zhixiong.li@Knights.ucf.edu

Dazhong Wu[1]
Department of Mechanical and Aerospace Engineering,
Department of Industrial Engineering and Management Systems,
University of Central Florida,
Orlando, FL 32816
e-mail: dazhong.wu@ucf.edu

Tianyu Yu
Department of Mechanical and Aerospace Engineering,
University of Central Florida,
Orlando, FL 32816
e-mail: tianyu.yu@ucf.edu

Chemical mechanical planarization (CMP) has been widely used in the semiconductor industry to create planar surfaces with a combination of chemical and mechanical forces. A CMP process is very complex because several chemical and mechanical phenomena (e.g., surface kinetics, electrochemical interfaces, contact mechanics, stress mechanics, hydrodynamics, and tribochemistry) are involved. Predicting the material removal rate (MRR) in a CMP process with sufficient accuracy is essential to achieving uniform surface finish. While physics-based methods have been introduced to predict MRRs, little research has been reported on monitoring and predictive modeling of the MRR in CMP. This paper presents a novel decision tree-based ensemble learning algorithm that can train the predictive model of the MRR. The stacking technique is used to combine three decision tree-based learning algorithms, including the random forests (RF), gradient boosting trees (GBT), and extremely randomized trees (ERT), via a meta-regressor. The proposed method is demonstrated on the data collected from a CMP tool that removes material from the surface of wafers. Experimental results have shown that the decision tree-based ensemble learning algorithm using stacking can predict the MRR in the CMP process with very high accuracy. [DOI: 10.1115/1.4042051]

Keywords: chemical mechanical planarization (CMP), material removal rate (MRR), predictive modeling, ensemble learning, stacking

1 Introduction

Chemical mechanical planarization (CMP) was invented in IBM in the early 1980s by Klaus D. Beyer to create a planar surface and enable subsequent lithographic imaging [1]. Since then the CMP process has been used to planarize various materials such as semiconductors, silicon oxide, metal, composites, and polymers with the combination of mechanical abrasion and chemical erosion [2]. CMP is one of the most important semiconductor processes, which is critical for producing microprocessors and memory chips. The global CMP market in 2014 was valued at $3.32 billion and is estimated to reach $4.94 billion by 2020 [3]. The key factors driving the growth of the CMP market are the increasing need of CMP for wafer planarization, high demand for consumer electronic products, and increasing use of micro-electro-mechanical systems.

A typical CMP tool consists of a rotating table used to carry a polishing pad, a replaceable polishing pad attached to the rotating table, a translating and rotating wafer carrier used to carry a wafer, a slurry dispenser, and a translating and rotating dresser used to condition the polishing pad. During the CMP process, a wafer is pressed against the polishing pad while the wafer carrier and a polishing pad are rotating in the same direction. Chemical slurries with abrasive particles are deposited onto the polishing pad during the CMP process. Modern CMP is a very complex process that involves several chemical and mechanical phenomena such as machinery kinetics, contact mechanics, hydrodynamics, chemical etching, and tribochemistry. The performance of the CMP process is measured using metrics such as material removal rate (MRR), planarization (e.g., within-wafer uniformity, wafer–wafer uniformity, and surface roughness), and process robustness and stability.

One of the key challenges in CMP is to achieve a high MRR and low nonuniformity of the polished surface simultaneously. Fundamental understanding of the material removal mechanism in CMP is critical to predict and control the quality of polished surfaces. Current methods for predicting the MRR in the CMP process fall into three categories: physics-based, model-based, and data-driven methods [3]. One of the most well-known physics-based models is the Preston equation [4]: $\mathrm{MRR} = K_p P^{\alpha} V^{\beta}$, where P denotes the downward pressure applied to a wafer, V denotes the relative rotating speed between the wafer and the polishing pad, K_p is the Preston coefficient, and α and β are the parameters depending on operating conditions. According to Krishnan et al. [1], the MRR of the CMP process is affected by many process variables such as the downward pressure, relative velocity between a polishing pad and a wafer, slurry flow rate, the usage of dresser, wafer hardness, pad hardness, pad roughness, and abrasive size. However, few physics-based and model-based methods are capable of predicting the MRR with sufficient accuracy by taking into account these process variables. For example, the limitation of physics-based methods is that the majority of these methods are developed based on the original and modified Preston equations [5–7], which are empirical models. The limitation of model-based methods is that certain distributions and assumptions made when developing close-form analytical solution do not always hold true [8,9]. Recent advances in artificial intelligence and machine learning enable predictive modeling of the complex CMP process by analyzing large volumes of condition monitoring data and identifying patterns [10–12]. Therefore, the objective of this study is to develop an ensemble learning approach to predicting the MRR of a wafer CMP process using large volumes of real-time condition monitoring data and a stacking technique.

The remainder of this paper is organized as follows: Section 2 reviews the related work on CMP. Section 3 introduces an ensemble learning-based predictive modeling approach using stacking. Section 4 presents a case study as well as discusses experimental results. Section 5 provides conclusions and future work.

[1]Corresponding author.

Manuscript received March 27, 2018; final manuscript received November 17, 2018; published online January 17, 2019. Assoc. Editor: Qiang Huang.

2 Related Work

Over the past decade, a few physics-based models based on the Preston equation have been introduced to predict the MRR in CMP. Lin and Wu [13] investigated the effects of relative velocity, downward pressure, the flow rate of flurry on MRR prediction. The experimental results have shown that the MRR increases as the downward pressure and relative velocity increase. The experimental results also suggested that the Preston equation could be further modified to improve prediction accuracy. Lee and Jeong [14] introduced a model that estimates the MRR for copper using a modified Preston equation. Three variables, including normal contact stress, relative velocity, and chemical reaction rate, were incorporated into the original Preston equation. Experimental results have shown that the modified Preston equation can estimate the MRR more accurately. Lee et al. [15] proposed a predictive model for MRR by taking into account the effects of the size, concentration, distribution of particles, slurry flow rate, polishing pad surface topography, and chemical reactions. Experimental results have shown that the estimated MRRs are in good agreement with the experimental data.

In addition to physics-based methods, model-based and data-driven methods have been introduced to predict MRR for CMP. Lih et al. [16] introduced a data-driven approach to the prediction of the MRR for CMP using an adaptive neuro-fuzzy inference system. Experimental results have shown that the predictive model trained by the adaptive neuro-fuzzy inference system outperforms that of artificial neural networks. Wang et al. [17] introduced a data-driven approach to the prediction of the MRR for CMP using a deep belief network. The particle swarm optimization was used to optimize the deep belief network structure and the learning rate. Experimental results have shown that an average root-mean-square error (RMSE) of 2.7 can be achieved. Jia et al. [18] proposed an adaptive method based on polynomial neural networks to predict the MRR for CMP. Experimental results have shown that the proposed method outperforms the k-nearest neighbors (KNN), logistic regression (LR), support vector regression (SVR), and random forests (RF) in terms of mean squared error (MSE) and coefficient of determination (R^2). Kong et al. [8] introduced a model-based method that integrates nonlinear Bayesian analysis and statistical methods to predict MRR, surface finish, and surface defects. The particle filtering method was used for nonlinear Bayesian analysis to predict the CMP process performance. A set of CMP experiments on copper wafers was conducted to collect vibration signals from a CMP machine. Experimental results have shown that the predictive model achieved a R^2 value of 0.96. Rao et al. [19] developed a deterministic process–machine interaction model that can identify complex time–frequency patterns during a CMP process for polishing copper wafer surfaces. The model was validated using a CMP machine instrumented with an accelerometer. Experimental results have shown that the model was able to predict MRR with a coefficient of determination of 0.85.

In summary, while previous research efforts have been focused on the development of physics-based and model-based predictive modeling techniques for CMP, these methods can take into account very few process variables. To address this research gap, the objective of this study is to develop a data-driven predictive modeling approach to predicting the MRR of CMP processes using large volumes of condition monitoring data.

3 Ensemble Learning-Based Predictive Modeling

Ensemble learning is a meta-algorithm that combines multiple machine learning algorithms (also known as base learners) into one learning algorithm to improve the performance of predictive models [20,21]. The base learners can be aggregated to reduce variance using randomization ensemble [5], to reduce bias using boosting ensemble [22], or to reduce both variance and bias using stacking ensemble. In general, the predictive model trained by ensemble learning outperforms that of individual base learners [23]. Table 1 lists three ensemble methods, including boosting, randomization, and stacking. None of these ensemble methods outperforms other methods consistently. However, some empirical studies have shown that the stacking method outperforms boosting and randomization [20]. Therefore, stacking is used to combine multiple base learners.

Figure 1 illustrates a computational framework of the ensemble learning-based predictive modeling approach. A training dataset is used to develop the predictive model. A validation dataset and a test dataset are used to validate and test the predictive model trained by the training dataset, respectively. The training, validation, and test datasets contain raw sensory data. A set of features in the time and frequency domains is extracted. To reduce the dimensionality of the features, RF is used to reduce the number of features based on a measure called variable importance. In the model training phase, the reduced subset of features is fed into the decision tree-based ensemble learning algorithm that combines three base learning algorithms, including RF, gradient boosting trees (GBT) [22], and extremely randomized trees (ERT) [24]. To develop a more accurate predictive model while avoiding overfitting, k-fold cross-validation (CV) is conducted to train the base learners. The outputs of the base learners are fed into another machine learning algorithm to train a meta-regressor. Two machine learning algorithms, including extreme learning machines (ELM) and classification and regression tree (CART), are used to train the meta-regressor. The output of the meta-regressor is the final prediction of the ensemble learning-based predictive modeling method. In the model validation and testing phases, the validation and test datasets are used to validate the predictive model.

3.1 Feature Extraction and Selection. In the feature extraction and selection phase, the raw measurement data were first transformed into a set of features in the time and frequency domains. Second, to reduce the dimensionality of the features, a subset of the original features was selected as input to the ensemble learning algorithm. Feature selection was conducted using RF based on variable/feature importance. The importance of a variable x_i for predicting a response variable Y is evaluated by averaging the sum of the weighted reduction in the residual sum of squares for all nodes t where x_i is used over the number of regression trees. The importance of a variable is given by

$$\text{VarImp}(x_i) = \frac{1}{N_T}\sum_{T}\sum_{t\in T: v(s_t)=x_i} p(t)\Delta k(s_t,t) \tag{1}$$

where $p(t)\Delta k(s_t,t)$ denotes the weighted reduction in residual sum of squares by splitting an internal node t into two child nodes. $p(t) = N_t/N$ denotes the proportion of the data points/samples at node t. N_t denotes the number of samples at node t. N denotes the total number of samples that is drawn to build a regression tree. N_T denotes the total number of regression trees in a random forest. T denotes a regression tree structure. s_t denotes a split at node t. $v(s_t)$ denotes the splitting variable that is selected for the split s_t. $\Delta k(s_t,t) = k(t) - p_1 k(t_1) - p_2 k(t_2)$. $k(t)$ denotes the residual sum of squares at node t. t_1 and t_2 denote the two child nodes of node t.

3.2 Base Learning Algorithms. In the model training phase, the ensemble learning algorithm combines three decision tree-based learning algorithms, including the RF, GBT, and ERT. The

Table 1 Comparison of different ensemble methods

Ensemble method	Objective	Ensemble type	Aggregation method
Boosting	Decrease bias	Sequential	Average
Randomization	Reduce variance	Parallel	Weighted average
Stacking	Both	Hybrid	Regression

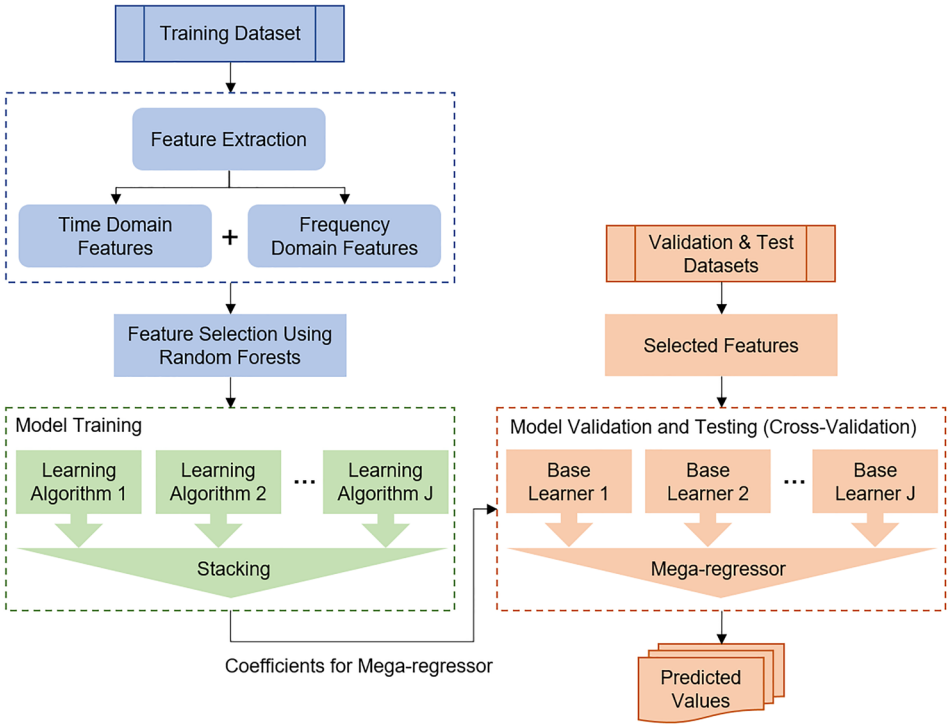

Fig. 1 A predictive modeling framework based on ensemble learning

three decision tree-based learning algorithms are briefly introduced in Secs. 3.2.1–3.2.3.

3.2.1 Random Forests. The RF algorithm [11,25–27] is an ensemble learning method that builds predictive models by combining multiple decision tree models. Each decision tree model is trained on a bootstrap sample generated from a training dataset using bootstrapping aggregating or bagging. Given a training dataset, bagging generates a set of new training samples by sampling from the original training dataset with replacement. By sampling with replacement, some observations may be repeated. However, bagging can reduce variance and avoid overfitting. In general, a RF model consists of a few hundred to several thousand decision trees. In this study, 100 regression trees were constructed. The second step of random forests is to construct a decision tree using an individual bootstrap sample. A bootstrap sample is the root node of an individual decision tree. To construct a decision tree, a node is split into two child nodes. To split a node, a set of variables is selected randomly. Selecting a subset of the variables instead of all of the variables can reduce the correlation of the decision trees. To determine an optimal split, one variable (also known as a splitting variable) is selected from a subset of the variables. The value of the splitting variable is referred to as a cutting point. In this study, one third of the total number of variables was selected. The splitting criterion at each node is to solve the following objective function:

$$\min_{j,s}\left[\min_{c1}\sum_{x_i\in R_1(j,s)}(y_i-c1)^2+\min_{c2}\sum_{x_i\in R_2(j,s)}(y_i-c2)^2\right] \quad (2)$$

where $j=1,2,\ldots,p$ (p denotes the number of splitting variables.) s denotes a cutting point. $R_1(j,s)=\{X|X_j\le s\}$ and $R_2(j,s)=\{X|X_j\ge s\}$ denote two resulting regions after the best split is determined. X_j denotes the jth splitting variable. $c1=\text{ave}\left(y_i|x_i\epsilon R_1(j,s)\right)$ denotes the average of the y_i's that fall into the region $R_1(j,s)$. $c2=\text{ave}\left(y_i|x_i\epsilon R_2(j,s)\right)$ denotes the average of the y_i's that fall into the region $R_2(j,s)$.

The splitting process is repeated until a stopping criterion is satisfied. In this study, the stopping criterion is satisfied when the number of data points in a node falls below a threshold of five. After 100 regression trees are constructed, a prediction at a new point can be made by averaging the predictions from all the regression trees.

3.2.2 Gradient Boosting Trees. As opposed to bagging, the GBT algorithm is an ensemble learning algorithm that develops decision tree-based predictive models sequentially using a boosting method [22]. In the boosting method, instances that are difficult to predict using the previous base learner appear more often in the training data than the ones that are correctly predicted. The key difference between bagging and boosting is that each instance is uniformly selected in bagging, whereas the probability that each instance is selected is not equal in boosting. Prediction accuracy is improved by assigning greater weights on the instances that are difficult to predict. The objective of GBT is to find an approximation to a function that minimizes a certain loss function. The predictor aims to estimate the response of an arbitrary input vector $\mathbf{x}$ using a mapping function $y=f(\boldsymbol{x})$ learned from a training set $(y_i, \mathbf{x}_i)$ ($i=1, 2, \ldots, N$ with N being the total number of training data). The mapping function $f(\cdot)$ [22,28] is to minimize the loss function $M(y_i,f(\mathbf{x_i}))$ as follows:

$$f^*=\arg\min_f\sum_{i=1}^{N}M(y_i,f(\mathbf{x_i}))=\arg\min_f E[M(y_i,f(\mathbf{x_i}))] \quad (3)$$

where $E[\cdot]$ denotes the expectation operation. The loss function could be the square error or negative binomial log-likelihood. A boosting model given by Eq. (4) is often used to approximate the mapping function

$$f(\mathbf{x})=\sum_{m=1}^{M}\beta_m h(\mathbf{x};\boldsymbol{\alpha_m}) \quad (4)$$

where $h(\mathbf{x};\boldsymbol{\alpha_m})$ ($m=1, 2, \ldots, M$) is a set of parameterized functions of $\mathbf{x}$, M is the number of functions, β_m is a weight coefficient, and $\boldsymbol{\alpha_m}=[\alpha_1, \alpha_2, \ldots]$ is a set of parameters that characterize the inputs in $h(\cdot)$. The greedy-stagewise approach [22] can be

used to solve the optimization problem given by Eq. (3). The parameters β_m and $\boldsymbol{\alpha_m}$ can be approximated by the below equation:

$$(\beta_m, \boldsymbol{\alpha_m}) = \arg\min_{\beta,\alpha} E\left[M\left(y_i, f_{m-1}(\boldsymbol{x_i}) + \beta h(\boldsymbol{x_i}; \boldsymbol{\alpha})\right)\right] \tag{5}$$

The boosting model $f(\mathbf{x})$ can be updated by the below equation:

$$f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \beta_m h(\mathbf{x}; \boldsymbol{\alpha_m}) \tag{6}$$

To solve the optimization problem in Eq. (5), the gradient boosting method adopts a two-step procedure. In the first step, the parameterized functions are fit by Eq. (7) along the best greedy step direction determined by Eq. (8)

$$\boldsymbol{\alpha_m} = \arg\min_{\alpha,\beta} E[(-\gamma_m(\mathbf{x_i}) - \beta h(\mathbf{x_i}; \boldsymbol{\alpha}))^2] \tag{7}$$

$$-\gamma_m(\mathbf{x_i}) = \left[\frac{\partial E[y_i, f(\mathbf{x_i})]}{\partial f(\mathbf{x_i})}\right]_{f(\mathbf{x})=f_{m-1}(\mathbf{x})} \tag{8}$$

In the second step, the optimal β_m is determined using the below equation:

$$\beta_m = \arg\min_{\rho} E[M\left(y_i, f_{m-1}(\mathbf{x_i}) + \beta h(\mathbf{x_i}; \boldsymbol{\alpha_m})\right)] \tag{9}$$

Thus, the mapping function can be updated by the below equation:

$$f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \beta_m h(\mathbf{x}; \boldsymbol{\alpha_m}) \tag{10}$$

Equations (2)–(9) briefly introduce the gradient boosting method in which the parameterized function can be wavelet or radial basis functions. If the parameterized function $h(\mathbf{x}; \boldsymbol{\alpha_i})$ is expressed using a regression tree, the parameter $\boldsymbol{\alpha_i}$ includes the splitting variables, cut points, and nodes of the individual trees.

3.2.3 Extremely Randomized Trees. The ERT algorithm builds an ensemble of unpruned decision trees from a complete learning sample rather than a bootstrap sample [24]. More importantly, a cut point is selected at random to split a node in ERT rather than choosing the best cut-point based on a local sample in RF. The ERT algorithm consists of the following three steps:

- *Step 1:* Determine three key parameters in an ERT model, namely, K (the number of randomly selected attributes at each node), $n_{\min}$ (the minimum sample size for splitting a node), and M (the number of trees in the ensemble model).
- *Step 2:* Build a decision tree on the original training data. The splitting criterion is to choose the best split among the K attributes that are randomly selected.
- *Step 3:* Construct an ensemble tree model by aggregating M totally randomized trees, which are acquired by repeating step 2 for M times.

3.3 Ensemble Learning Using Stacking. Stacking is a method that combines multiple classification or regression models using a meta-classifier or a meta-regressor. An ensemble learning method using stacking includes two steps: (1) training base learners and (2) training a meta-regressor. Figure 2 illustrates the two-layer ensemble learning method using stacking. Pseudocode that describes the ensemble learning method using stacking can be found in Table 2.

4 Case Study

In this section, the decision tree-based ensemble learning method using stacking is demonstrated on the dataset acquired from the 2016 PHM data challenge [29].

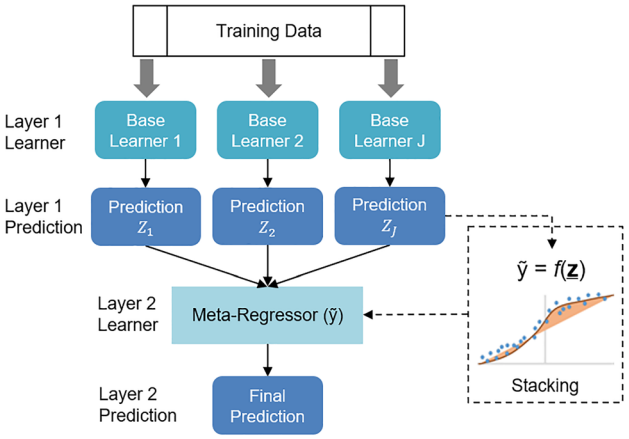

Fig. 2 Two-layer ensemble learning using stacking

Table 2 Pseudocode of the ensemble learning algorithm using stacking

Input:	Training dataset $\mathbf{S} = \{y_i, \mathbf{x_i}\}_{i=1}^{N}$ ($\mathbf{x_i}$: Input variable; y_i: Response variable)
1:	Train the base learners at layer 1 using S
2:	**for** $j = 1$ to J (J: Number of base learning algorithms)
3:	build base learner d_j using base learning algorithm j
4:	**end for**
5:	Generate new training data $\mathbf{R}$
6:	**for** $i = 1$ to N do
7:	$R_i = \{y_i, \mathbf{z_i}\}$, where $\mathbf{z_i} = [d_1(\mathbf{x_i}), d_1(\mathbf{x_i}), \ldots, d_J(\mathbf{x_i})]$
8:	**end for**
9:	Build a mega-regressor D using $\mathbf{R}$ at layer 2
Output:	Predictive model D

4.1 Data Description. The dataset contains multiple sensory signals collected from a CMP tool that removes the material from wafers. The total volume of the dataset is 187 MB.

Figure 3 shows a schematic diagram of the CMP process. The CMP tool consists of a rotating table, a replaceable polishing pad, a rotating wafer carrier, a slurry dispenser, and a dresser. A wafer is placed on the underside of the wafer carrier. During the CMP process, the wafer is pressed against the polishing pad while the polishing pad and wafer carrier are rotating in the same direction. A slurry composed of abrasive materials and chemicals is dispensed onto the polishing pad during the CMP process. The dresser is composed of a hard material such as diamond that is

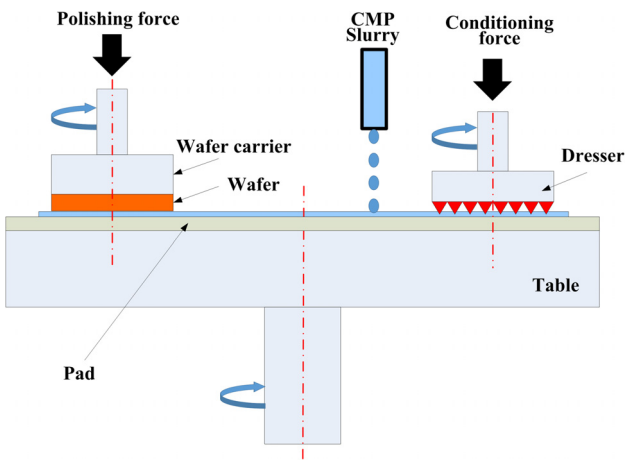

Fig. 3 Schematic diagram of the CMP process

pressed across the polishing pad to roughen the surface of the polishing pad.

The dataset includes 19 variables. Table 3 lists the description of these measurement variables and additional six parameters. The conditions of four CMP tools were monitored during various runs of the CMP tools for specified wafers. Some variables, including the usage of the polish-pad backing film, dresser, polishing table, dresser table, polishing membrane, wafer carrier sheet, and the status of the dressing water, are real-time condition monitoring data. Other variables such as the pressure-related variables, slurry flow rate, the rotating rate of the wafer, stage, and head, are preset to different values in different tests. The raw data are divided into three datasets, including one training dataset, one validation dataset, and one test dataset. Table 4 provides more details on the three datasets. The training data were collected from 1981 wafers in two operational stages: A and B. A total number of 672,744 trajectories were collected from the wafers. These training trajectories are stored in 185 files in CSV format. The validation and test datasets include 144,148 and 156,262 trajectories, respectively. It should be noted that four outliers in the training dataset under stage A were removed before processing the data [18,30].

The predictive model was trained on the training dataset, and then validated on the validation and test datasets. The validation and test datasets provide an unbiased evaluation of the model fit on the training dataset.

4.2 Feature Extraction and Selection. The raw data were transformed into a set of features and then a reduced subset of features before being processed by the decision tree-based ensemble learning algorithm. Four statistical features (see Eqs. (11)–(14)) in the time domain, including the standard deviation, central moment, skewness, and kurtosis, were extracted from each sensor signal. In addition, other three features in the frequency domain, including the maximum frequency amplitude, frequency center, and kurtosis of frequencies, were extracted from the measurement variables x_{21}, x_{22}, and x_{23}. Eighty-five (85) features in total were extracted from the raw data

$$\text{standard deviation} \quad \sigma(\mathbf{x}) = E[\mathbf{x} - \mu]^{1/2} \tag{11}$$

$$\text{central moment} \quad m_p(\mathbf{x}) = E[\mathbf{x} - \mu]^p \tag{12}$$

$$\text{skewness} \quad s(\mathbf{x}) = E[\mathbf{x} - \mu]^3/\sigma^3 \tag{13}$$

$$\text{kurtosis} \quad k(\mathbf{x}) = E[\mathbf{x} - \mu]^4/\sigma^4 \tag{14}$$

where $E[\cdot]$ denotes the expectation operation, μ is the mean value of $\mathbf{x}$, and p is the order of moment. In this study, the central moment of order 3 is used ($p = 3$).

Table 4 Training, validation, and test datasets

Information	2016 PHM CMP datasets		
	Training	Validation	Test
Total number of observations	672,744	144,148	156,262
Number of wafers	1,981	424	424
Number of wafers under stage A	1,166	252	238
Number of observations under stage A	376,859	82,984	91,798
Number of wafers under stage B	815	172	186
Number of observations under stage B	295,885	61,164	64,464

To avoid overfitting and reduce training time, a subset of the extracted 85 features was selected for use in model development. A feature selection algorithm can be considered as a search technique that evaluates the importance of individual features. As shown in Fig. 4, RF was used to measure the importance of each feature. When the threshold value is set equal to 0.65, the top five most important features include (1) the standard deviation of the slurry flow rate, (2) the standard deviation of the usage of the polish-pad backing film, (3) the skewness of the downward pressure, (4) the central moment of the usage of the polishing table, and (5) the central moment of the usage of the wafer carrier sheet. The ranking of the variable importance makes sense from a physics point of view. For example, the slurry flow rate has a significant impact on the amount of active abrasives in the CMP process, thereby affecting the MRR significantly. The usage of polishing-pad backing film and wafer carrier sheet has an impact on the local contact pressure between the polishing pad and the wafer, thereby affecting the MRR [31,32]. The downward pressure has an impact on the MRR according to the Preston equation. The usage of the polishing table has an impact on pad wear, which dynamically changes the pad surface topology in the CMP process, thereby affecting the MRR.

The number of selected features was determined by balancing the trade-off between prediction accuracy and training time. Prediction accuracy is measured using R-square (R^2), RMSE, relative error (RE), and score function (S-score) (see Eqs. (15)–(18)). RMSE and RE measure the deviations between the predicted and actual MRRs. R^2 measures the goodness of fit of a predictive model. An S-score, initially introduced in 2008 PHM data challenge, measures the performance of a model by taking into account whether the model overestimates and underestimates the MRR

$$\text{RMSE} \quad \varepsilon_{\text{RMSE}} = \sqrt{E[(\hat{\mathbf{y}} - \mathbf{y})^2]} \tag{15}$$

$$\text{RE} \quad \varepsilon_{\text{RPE}i} = |\hat{y}_i - y_i|/y_i \tag{16}$$

Table 3 Data description

Symbol	Description	Symbol	Description
x_1	Machine ID	x_{14}	Pressure applied to the retainer ring
x_2	Wafer ring location ID	x_{15}	Pressure applied to the ripple air bag
x_3	Time (s)	x_{16}	Usage of polishing membrane
x_4	Wafer ID	x_{17}	Usage of wafer carrier sheet
x_5	Stage ID (A or B)	x_{18}	Flow rate of slurry type A
x_6	Chamber ID	x_{19}	Flow rate of slurry type B
x_7	Usage of polish-pad backing film	x_{20}	Flow rate of slurry type C
x_8	Usage of dresser	x_{21}	Rotating rate of wafer
x_9	Usage of polishing table	x_{22}	Rotating rate of stage
x_{10}	Usage of dresser table	x_{23}	Rotating rate of head
x_{11}	Chamber pressure	x_{24}	Status of dressing water
x_{12}	Pressure applied to the main outer air bag	x_{25}	Pressure applied to the edge air bag
x_{13}	Pressure applied to the center air bag		

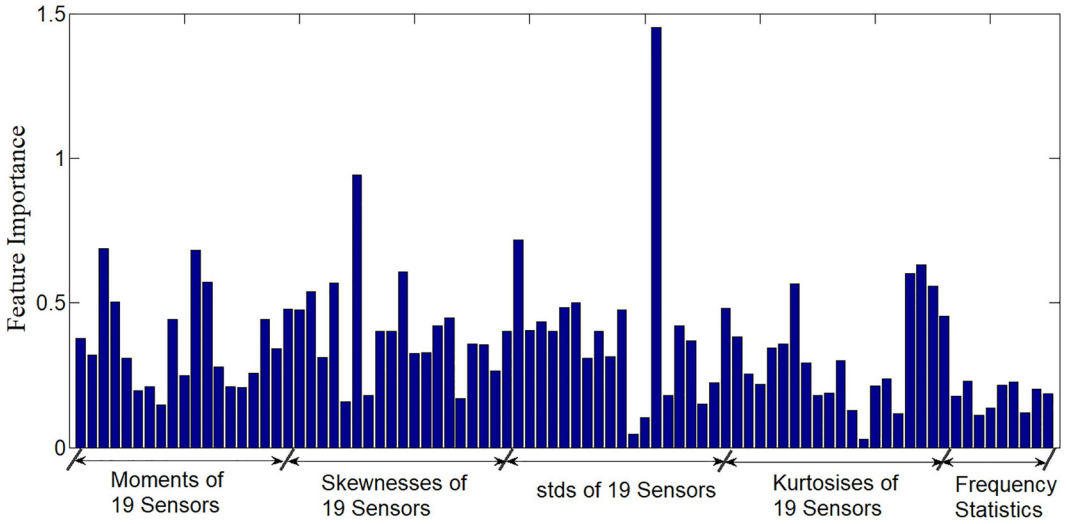

Fig. 4 Variable importance of the extracted 85 features

$$\text{S-score} \qquad \varepsilon_{\text{CV}i} = \begin{cases} \exp(-d_i/13), d_i < 0 \\ \exp(d_i/10), \quad d_i \geq 0 \end{cases}, (d_i = \hat{y}_i - y_i) \tag{17}$$

$$\text{R-square}(R^2) \qquad \begin{cases} \varepsilon_{R^2} = 1 - SR/ST \\ SR = \sum_i \hat{y}_i - \bar{y}^T \\ ST = \sum_i \hat{y}_i - y_i^T \end{cases} \tag{18}$$

where $i = 1, 2, \ldots, N$ (N is the sample number), y_i is the actual MRR of the ith sample, $\hat{y}_i$ is the predicted MRR of the ith sample, $\hat{\mathbf{y}}$ is the matrix form of all the predicted MRRs, and $\bar{y}$ is the mean value of the actual MRR vector $\mathbf{y}$. To determine an optimal number of features, GBT, RF, and ERT were used to train predictive models using 5, 20, 35, 50, 65, and 85 features. The parameter settings for the three base learners are listed in Table 5. These decision tree-based learning algorithms were used to train predictive models using the training dataset. The validation dataset was used to evaluate the performance of the predictive models. All the computational experiments were conducted on a computer with Intel Core i7-6650 U CPU at 2.2 GHz and 16 GB of memory in the Windows 10 environment. To take into account computational uncertainty, we replicate a computational experiment for 20 times.

Figure 5 shows the average of R^2, RE, S-score, RMSE, and training time versus a varying number of features. As shown in Fig. 5(*a*), R^2 increases as the number of features increases for both GBT and RF. R^2 decreases as the number of features exceeds 50 for ERT. As shown in Fig. 5(*b*), RE decreases as the number of features increases for both GBT and RF. RE increases as the number of features exceeds 35 for ERT. As shown in Fig. 5(*c*), S-score decreases as the number of features increases for both GBT and RF. S-score increases as the number of features increases for ERT. As shown in Fig. 5(*d*), RMSE decreases as the number of features increases for both GBT and RF. RMSE increases as the number of features exceeds 35 for ERT. As shown in Fig. 5(*e*), training time almost does not vary with the number of features for both GBT and RF. However, training time increases as the number of features increases for ERT. By balancing the trade-off between prediction accuracy (i.e., R^2, RE, S-score, RMSE) and computational efficiency (i.e., training time), the optimal number of features is 35, which takes nearly minimum training time while achieving sufficient prediction accuracy. In addition to the average values of R^2, RE, S-score, RMSE, and training time, the variability of these performance measures are also calculated. Figure 6 shows a boxplot indicating the variability of RMSEs with regard to GBT, RF, and ERT algorithms using a varying number of features. The boxplot shows the median, minimum, and maximum RMSEs of 20 replications. As shown in Fig. 6, the variability of RMSEs is relatively small.

Figures 7–9 show the prediction performance on the validation dataset using the GBT, RF, and ERT methods and 35 features, respectively. For example, Fig. 7(*a*) shows the comparison between the predicted MRR and the actual MRR (i.e., the ground-truth MRR) in order of wafer index with regard to the GBT method. Figure 7(*b*) shows the comparison between the predicted MRR and the actual MRR (i.e., the ground-truth MRR) in order of MRR. The average R^2 is 0.917. Figure 7(*c*) shows the distribution of the residuals (i.e., the difference between the predicted and actual MRR). The standard deviation of the residuals is 8.317 nm/min. Similarly, Figs. 8(*a*) and 8(*b*) show the comparison between the predicted MRR and the actual MRR for the RF method. The average R^2 is 0.917. Figure 8(*c*) shows the standard deviation of the residues as 8.291 nm/min. Figures 9(*a*) and 9(*b*) show the comparison between the predicted MRR and the actual MRR for the ERT method. The average R^2 is 0.919. Figure 9(*c*) shows the standard deviation of the residues as 8.236 nm/min.

Figures 10–12 show the prediction performance on the validation dataset using the GBT, RF, and ERT methods and 85 features, respectively. For example, Fig. 10(*a*) shows the comparison between the predicted MRR and the actual MRR in order of wafer index for the GBT method. Figure 10(*b*) shows the comparison between the predicted MRR and the actual MRR in order of MRR. The average R^2 is 0.942. Figure 10(*c*) shows the standard deviation of the residuals is 6.909 nm/min. Figures 11(*a*) and 11(*b*) show the comparison between the predicted MRR and the actual MRR for the RF method. The average R^2 is 0.916. Figure 11(*c*) shows the standard deviation of the residues as 8.295 nm/min. Figures 12(*a*)–12(*c*) show the results for the ERT method. The average R^2 is 0.767. The standard deviation of the

Table 5 Parameter settings for the base learning algorithms

Base learner	Parameters
GBT	Number of trees = 100 and Number of leaves for each tree = 30
RF	Number of trees = 100
ERT	Number of trees = 100, Number of attributes at each node = 3 and Minimum sample size = 3

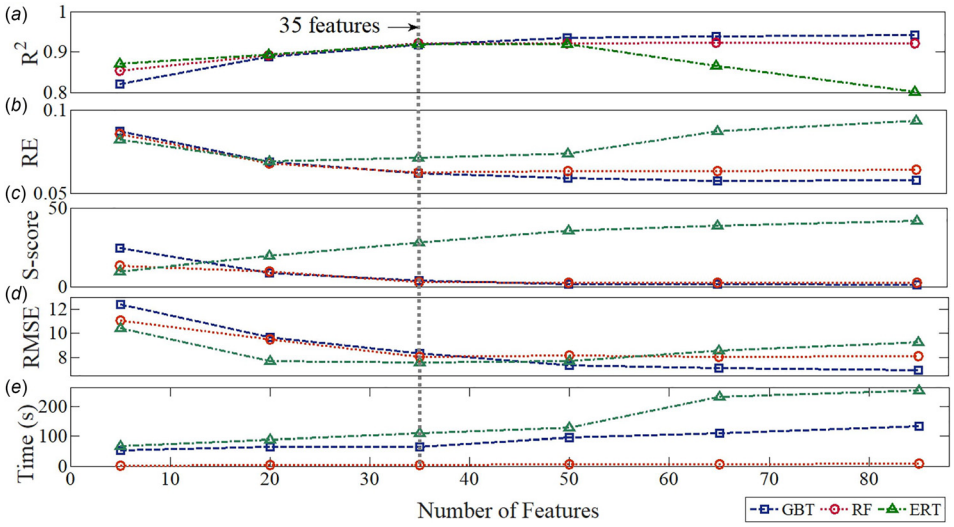

Fig. 5 Prediction performance versus a varying number of features: (*a*) R^2, (*b*) RE, (*c*) S-score, (*d*) RMSE, and (*e*) training time

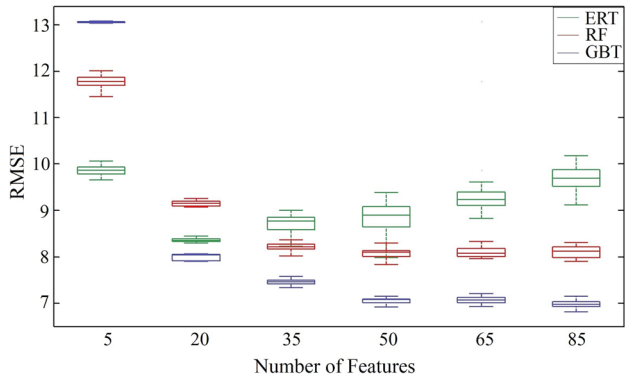

Fig. 6 Variability of RMSE for GBT, RF, and ERT algorithms

residues is 13.985 nm/min. The average R^2 and standard deviation of the predicted MRR using 35 features are comparable with those using 85 features. However, the amount of time spent training the predictive model using 35 features (180.998 s) is much shorter than that of 85 features (394.297 s).

4.3 Prediction Results Using Ensemble Learning. The data points in the training dataset were transformed into 35 features, and then fed into the decision tree-based ensemble learning algorithm as input. The predictive models trained by the ensemble learning methods were validated on the validation and test datasets.

Table 6 lists the R^2, RE, S-score, RMSE values for CART-based stacking and ELM-based stacking methods. The experimental results have shown that the decision tree-based ensemble learning methods using CART and ELM as stacking methods outperform the base learners. For the validation dataset, the ensemble learning method using CART outperforms the ensemble learning method using ELM in terms of R^2, RE, S-score, and RMSE. For the test dataset, the ensemble learning method using CART still

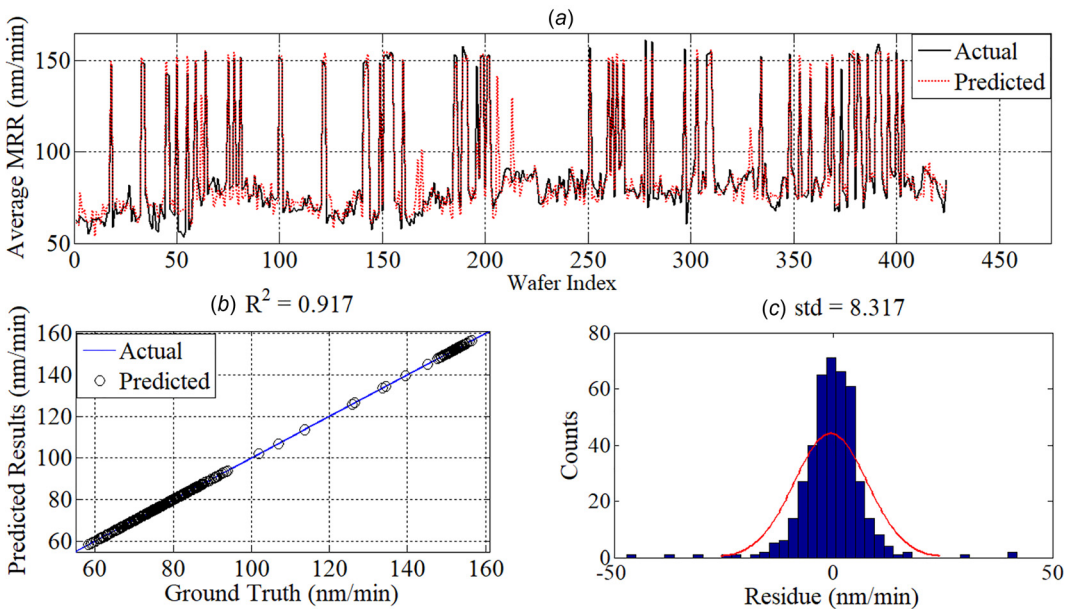

Fig. 7 Prediction performance on the validation dataset using GBT and 35 features

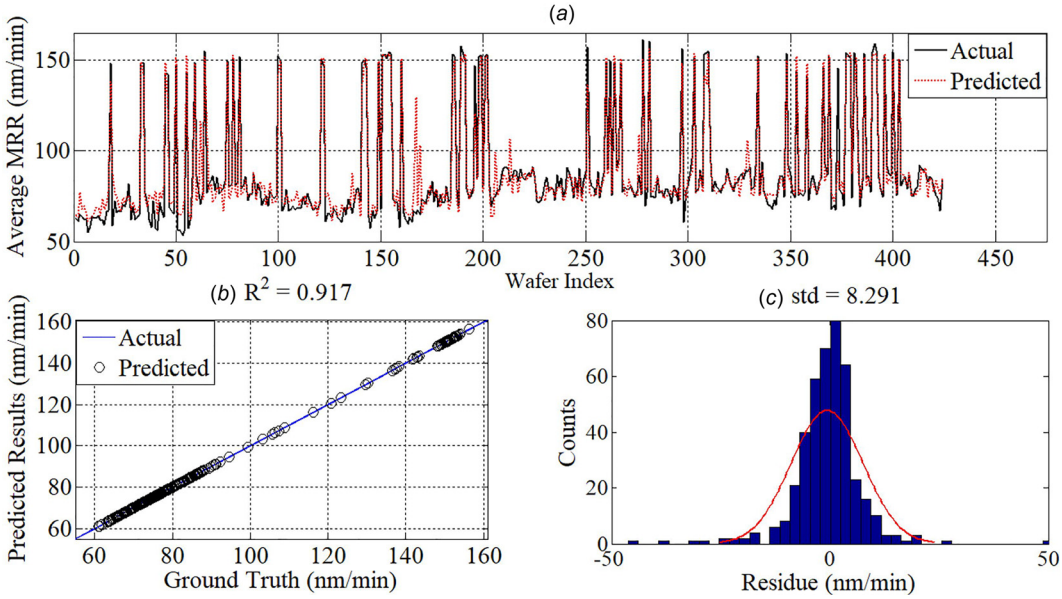

Fig. 8 Prediction performance on the validation dataset using RF and 35 features

outperforms the ensemble learning method using ELM in terms of R^2, S-score, and RMSE. However, the ensemble learning method using ELM outperforms the ensemble learning method using CART slightly in terms of RE.

Figure 13 shows a comparison between the CART- and ELM-based ensemble learning methods for the cases where 5, 20, 35, 50, 65, and 85 features were selected. Selecting 35 features achieves near-optimal performance in terms of R^2, RE, S-score, RMSE for the validation and test datasets. The result of this comparative study is consistent with that of the comparative study where the performance of GBT, RF, and ERT were compared using different number of features.

4.4 Accuracy Improvement With Stage Information. The CMP data were collected under two different stages: stages A and B [30]. Figures 14 and 15 show the prediction results by taking into account the stage information. As shown in Fig. 14, the standard deviation of the residuals for stage A using the CART- and ELM-based stacking methods is 3.994 nm/min and 4.215 nm/min, respectively. As shown in Fig. 15, the standard deviation of the residues for stage B using the CART- and ELM-based stacking methods is 4.088 nm/min and 4.417 nm/min, respectively.

Table 7 provides more details on the performance of the predictive models trained on the validation and test datasets for stage A using the GBT, RF, ERT, CART-, and ELM-based ensemble learning methods. The ensemble learning methods using CART and ELM outperform the base learners in terms of R^2, RE, S-score, RMSE for both validation and test datasets. CART-based stacking outperforms ELM-based stacking for the validation dataset. However, ELM-based stacking outperforms CART-based stacking for the test dataset.

Table 8 provides more details on the performance of the predictive models trained on the validation and test datasets for stage B

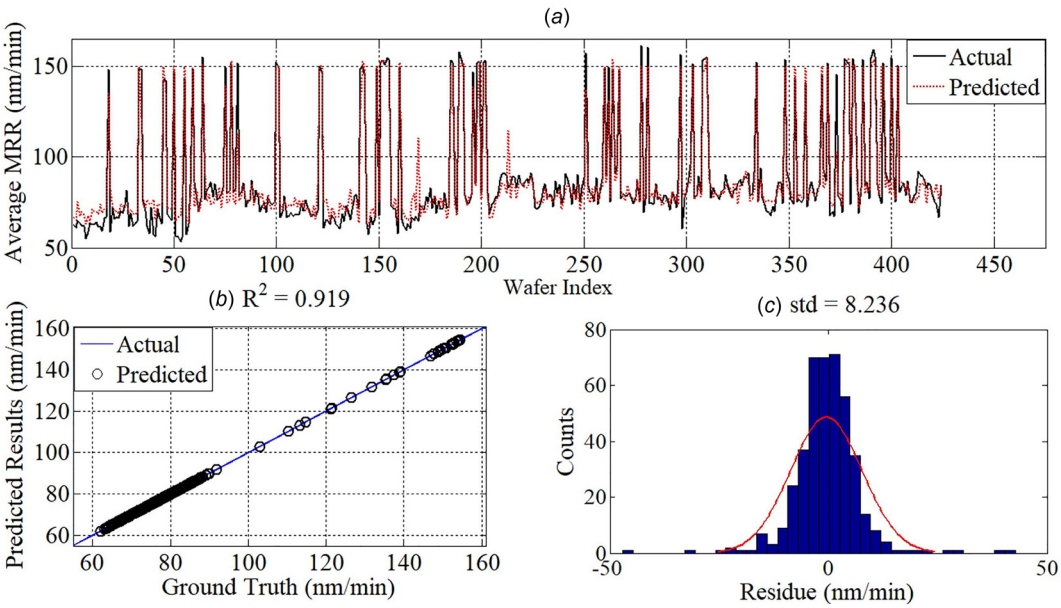

Fig. 9 Prediction performance on the validation dataset using ERT and 35 features

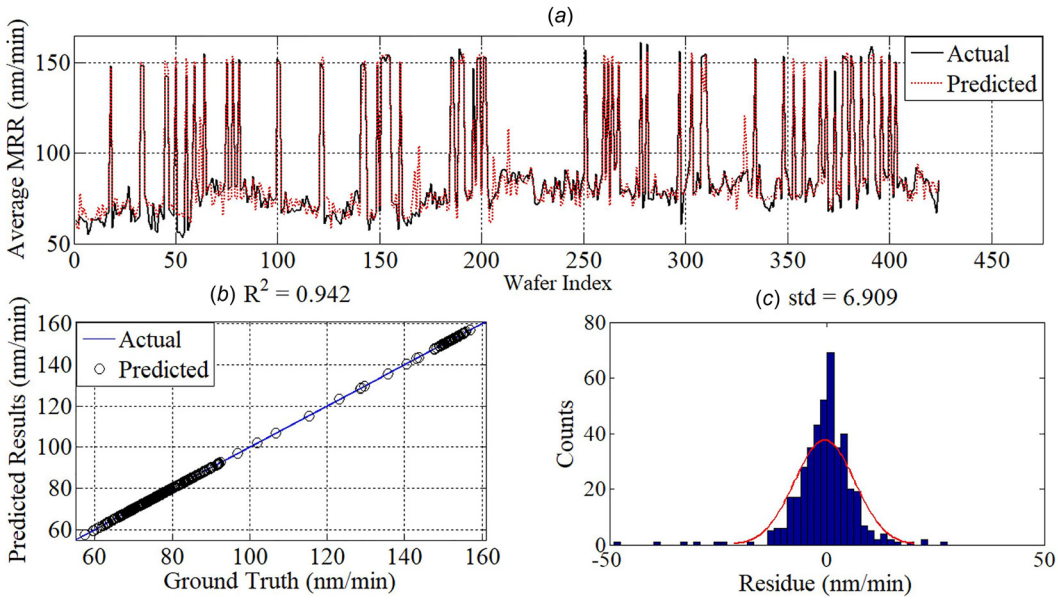

Fig. 10 Prediction performance on the validation dataset using GBT and 85 features

using the GBT, RF, ERT, CART-, and ELM-based ensemble learning methods. Similar to stage A, the ensemble learning methods using CART and ELM outperform the base learners in terms of R^2, RE, S-score, RMSE for both validation and test datasets. CART-based stacking outperforms ELM-based stacking for the validation dataset. However, ELM-based stacking outperforms CART-based stacking for the test dataset.

4.5 Discussions

4.5.1 Prediction Accuracy and Computational Efficiency. Although the training process in this study is off-line, the method we developed could be applied for online process monitoring where the predictive model of the MRR has to be retrained when some of the dominant operating parameters change significantly. Online monitoring requires effective and computationally efficient algorithms. In addition, while only several CMP tools were monitored in this study, large volumes of real-time condition monitoring data will be generated when hundreds of CMP tools are monitored in real time. In this case, it is important to evaluate both prediction accuracy and computational efficiency of the proposed algorithm. Moreover, prediction accuracy after reducing the dimensionality of the feature space will also be discussed.

It should be noted that the prediction accuracy is affected by the number of trees in the base learners. To evaluate the effect of the number of trees, a comparative study was conducted using different number of trees. Tables 9–11 summarize the errors of the predictive models trained by GBT, RF, and ERT and training time using 50, 100, 200, 400, and 800 trees, respectively. As shown in Table 9, R^2 and RE values do not vary significantly with the number of trees for GBT. RMSE and S-score values vary with the number of trees. For example, the RMSE is 6.815 for the predictive model trained by 100 trees, while the RMSE is 7.350 for the predictive model training by 50 trees. The S-score value is 0.953 for the predictive model trained by 100 trees, while the S-score value is 2.067 for the predictive model trained by 50 trees. When

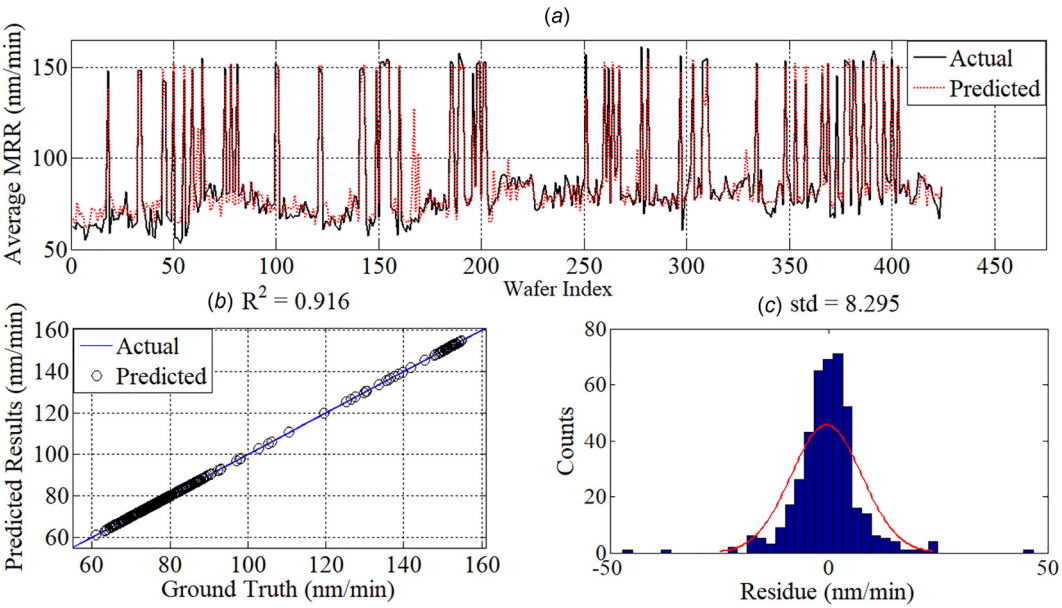

Fig. 11 Prediction performance on the validation dataset using RF and 85 features

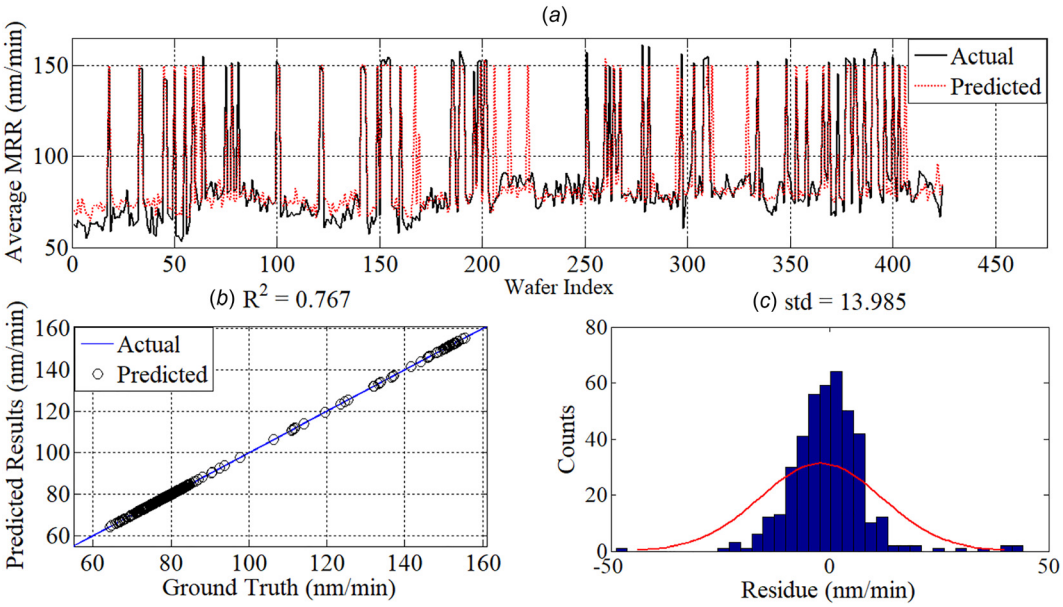

Fig. 12 Prediction performance on the validation dataset using ERT and 85 features

Table 6 Prediction performance of the ensemble learning algorithm for the validation and test datasets

		Prediction performance			
Dataset	Method	R^2	RMSE	RE	S-score
Validation dataset	GBT	0.917	8.323	0.062	3.915
	RF	0.917	8.066	0.063	2.476
	ERT	0.919	7.571	0.064	5.521
	CART-stacking	0.937	6.926	0.052	1.318
	ELM-stacking	0.905	7.222	0.057	3.452
Test dataset	GBT	0.919	8.252	0.058	2.224
	RF	0.918	8.572	0.063	6.876
	ERT	0.939	7.336	0.055	1.723
	CART-stacking	0.941	7.009	0.056	1.034
	ELM-stacking	0.940	7.261	0.054	2.051

the number of trees exceeds 100, the accuracy of the predictive models does not significantly improve. However, as shown in Fig. 16(*a*), the training time increases significantly with the number of trees. For example, the training times are 36.910, 132.598, 216.609, 468.120, and 973.425 s for 50, 100, 200, 400, and 800 trees, respectively.

Similarly, as shown in Table 10, R^2 and RE values do not vary significantly with the number of trees for RF. RMSE and S-score values vary with the number of trees. For example, the RMSE is 7.969 for the predictive model trained by 100 trees, while the RMSE is 8.218 for the predictive model training by 50 trees. The S-score value is 2.297 for the predictive model trained by 100 trees, while the S-score value is 2.568 for the predictive model trained by 50 trees. When the number of trees exceeds 100, the accuracy of the predictive models does not significantly improve. However, the training time increases significantly with the number of trees. For example, the training times are 5.490, 7.583,

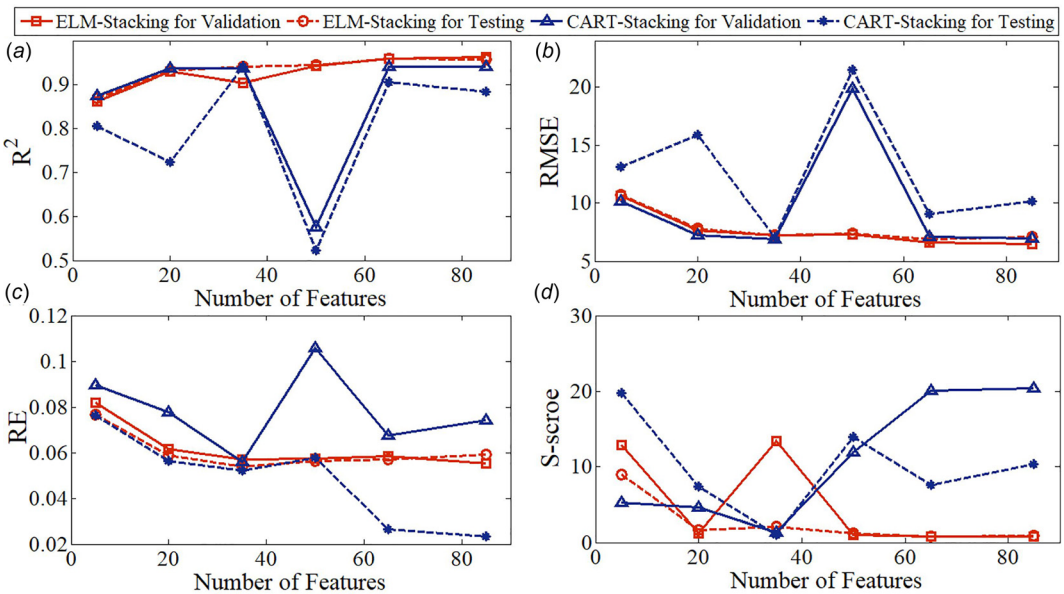

Fig. 13 Stacked ensemble results using different number of features: (*a*) R^2, (*b*) RE, (*c*) S-score, and (*d*) RMSE

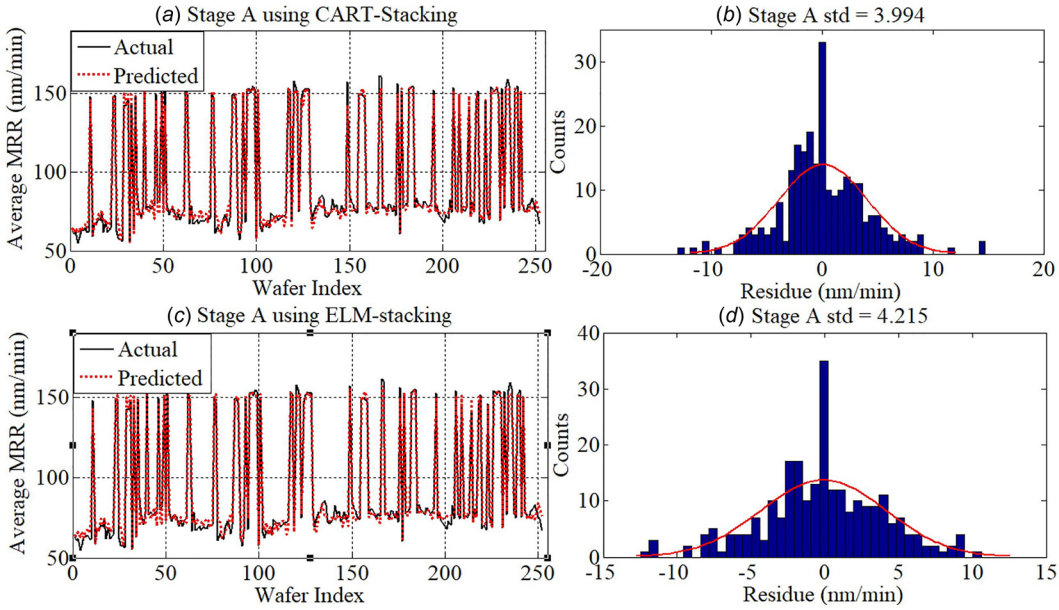

Fig. 14 Prediction performance for the validation dataset in stage A using the stacked ensemble: ((*a*) and (*b*)) stacking-CART and ((*c*) and (*d*)) stacking-ELM

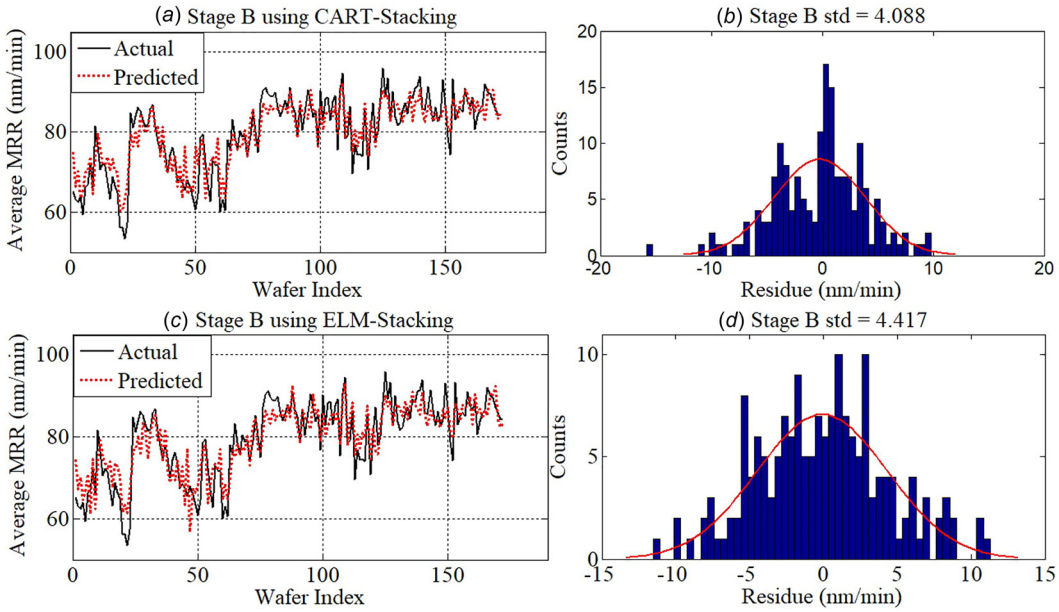

Fig. 15 Prediction results for the validation dataset in stage B using the stacked ensemble: ((*a*) and (*b*)) stacking-CART and ((*c*) and (*d*)) stacking-ELM

Table 7 Performance of the predictive model for stage A

		Prediction performance			
Dataset	Method	R^2	RMSE	RE	S-score
Validation dataset	GBT	0.977	5.415	0.043	0.500
	RF	0.976	5.514	0.045	0.511
	ERT	0.973	5.920	0.046	0.564
	CART-stacking	0.987	3.987	0.035	0.335
	ELM-stacking	0.986	4.208	0.037	0.370
Test dataset	GBT	0.970	6.552	0.051	1.674
	RF	0.981	5.395	0.046	0.512
	ERT	0.966	7.048	0.050	3.723
	CART-stacking	0.983	5.065	0.047	0.479
	ELM-stacking	0.984	4.795	0.043	0.446

Table 8 Performance of the predictive model for stage B

		Prediction performance			
Dataset	Method	R^2	RMSE	RE	S-score
Validation dataset	GBT	0.721	5.032	0.052	0.493
	RF	0.734	5.059	0.051	0.490
	ERT	0.763	4.947	0.051	0.487
	CART-stacking	0.826	4.082	0.041	0.370
	ELM-stacking	0.788	4.405	0.047	0.410
Test dataset	GBT	0.687	4.781	0.047	0.440
	RF	0.722	4.598	0.045	0.419
	ERT	0.701	4.792	0.048	0.444
	CART-stacking	0.725	4.500	0.044	0.404
	ELM-stacking	0.727	4.485	0.044	0.404

Table 9 Prediction performance of GBT using different number of trees

Number of features	Error	Number of trees				
		50	100	200	400	800
85	R^2	0.935	0.944	0.942	0.944	0.943
	RMSE	7.350	6.815	6.943	6.796	6.880
	RE	0.059	0.057	0.058	0.057	0.057
	S-score	2.067	0.953	1.105	1.050	1.064
	Training time (s)	36.910	132.598	216.609	468.120	973.425
35	R^2	0.916	0.919	0.917	0.909	0.908
	RMSE	8.340	8.213	8.329	8.701	8.763
	RE	0.062	0.062	0.062	0.064	0.064
	S-score	3.583	2.826	4.473	3.767	3.789
	Training time (s)	27.331	66.064	164.849	344.392	792.356

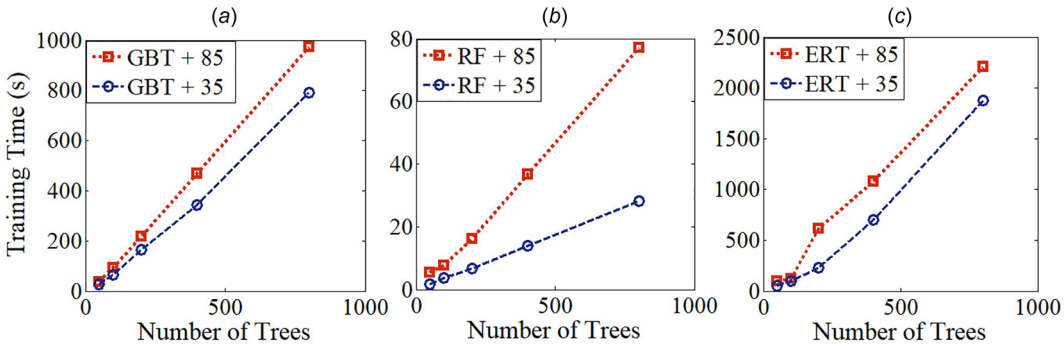

Fig. 16 Training time of base learners using different number of trees

Table 10 Prediction performance of RF using different number of trees

Number of features	Error	Number of trees				
		50	100	200	400	800
85	R^2	0.919	0.924	0.922	0.920	0.923
	RMSE	8.218	7.969	8.095	8.167	8.028
	RE	0.065	0.064	0.064	0.064	0.064
	S-score	2.568	2.297	2.230	2.319	2.272
	Training time (s)	5.490	7.791	16.190	36.636	77.130
35	R^2	0.917	0.921	0.921	0.925	0.9210
	RMSE	8.309	8.086	8.076	7.884	8.1007
	RE	0.063	0.062	0.062	0.062	0.0624
	S-score	2.725	2.816	2.599	2.316	2.6722
	Training time (s)	1.718	3.952	6.596	13.844	28.106

Table 11 Prediction performance of ERT using different number of trees

Number of features	Error	Number of trees				
		50	100	200	400	800
85	R^2	0.882	0.868	0.872	0.850	0.831
	RMSE	9.992	10.542	10.394	11.193	11.851
	RE	0.083	0.084	0.083	0.085	0.087
	S-score	3.197	9.687	8.926	23.399	26.319
	Training time (s)	96.901	253.908	616.882	1077.117	2207.526
35	R^2	0.914	0.901	0.902	0.892	0.884
	RMSE	8.485	9.056	9.022	9.480	9.817
	RE	0.067	0.069	0.067	0.069	0.070
	S-score	2.552	9.240	9.103	10.269	14.216
	Training time (s)	51.296	110.982	230.290	701.165	1876.672

Table 12 Comparison of the RMSE on the test dataset

	Preston's equation [18,30]	Luo–Dornfeld model [18,30]	CART-stacking	ELM-stacking
Stage A	42.3	7.6	5.065	4.795
Stage B	16.6	NA	4.500	4.485
Average	29.5	7.6	4.783	4.640

16.190, 36.636, and 77.130 s for 50, 100, 200, 400, and 800 trees, respectively. In comparison with GBT, RF is more computationally efficient.

Similar to GBT and RF, R^2 and RE values do not vary significantly with the number of trees for ERT as shown in Table 11. RMSE and S-score values increase with the number of trees. For example, the RMSE is 10.542 for the predictive model trained by 100 trees, while the RMSE is 9.992 for the predictive model training by 50 trees. The S-score value is 9.687 for the predictive model trained by 100 trees, while the S-score value is 3.197 for the predictive model trained by 50 trees. The training time increases significantly with the number of trees. For example, the training times are 96.901, 253.908, 616.882, 1077.117, and 2207.526 s for 50, 100, 200, 400, and 800 trees, respectively.

As shown in Tables 9–11, the prediction accuracy of GBT, RF, and ERT is comparable. However, RF is the most computationally efficient, while ERT is the least computationally efficient as shown in Fig. 16. To predict the MRR with sufficient accuracy while maintaining sufficient computational efficiency, the number of trees and number of features were set to 100 and 35, respectively.

To compare the performance of the proposed ensemble learning-based predictive modeling approach with some of the well-known models, a comparative study was conducted. Table 12 shows a comparison of four approaches, including Preston's equation, Luo–Dornfeld model, CART-based and ELM-based ensemble learning algorithms. Both CART-based and ELM-based ensemble learning algorithms outperform the Preston's equation and Luo–Dornfeld model significantly.

4.5.2 Meta-Regressor Selection. In addition to the number of trees, the selection of the meta-regressor also affects prediction accuracy. In this section, the performance of CART-based and ELM-based meta-regressors is compared with several other meta-regressors, including linear regression (LR) [33], Bayesian logistic regression (BLR) [34], SVR [35], and AdaBoost [36].

Table 13 summarizes the comparison of the results. The CART-based stacking method outperforms other stacking methods for the validation dataset in terms of RMSE. The ELM-based stacking method outperforms other stacking methods for the test dataset in terms of RMSE. The training times for the LR-, BLR-, AdaBoost-, SVR-, CART- and ELM-based stacking methods are 0.783, 44.734, 38.943, 84.763, 0.002, and 0.580 s, respectively. Among all of these meta-regressors, the CART- and ELM-based stacking methods are the most computational efficient based on the training time.

5 Conclusions and Future Work

This paper has presented an ensemble learning-based prognostic approach to predicting the MRR in the CMP process. A two-layer stacking ensemble learning technique was used to combine three decision tree-based machine learning algorithms, including GBT, RF and ERT. RF was also used to select the most important features. Two stacking techniques were used to combine RF, GBT, and ERT. This new method was demonstrated on the datasets acquired from the 2016 PHM data challenge. The predictive model was developed on a training dataset, and then was validated on the validation and test datasets. The performance metrics include R^2, RE, S-score, RMSE, and training time. The experimental results have shown that the decision tree-based ensemble learning approach predicts the MRR of the CMP process with sufficient accuracy and reasonable training time. In addition, the ensemble learning algorithm outperformed the base learners (i.e., RF, GBT, and ERT). In the future, the training process of the ensemble learning-based prognostics approach will be parallelized to improve the computation efficiency.

Acknowledgment

The research reported in this paper is partially supported by the University of Central Florida (UCF) and the Digital Manufacturing and Design Innovation Institute (DMDII). Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the UCF and the DMDII.

Table 13 Comparison of the RMSE using different mega-regressors

		LR-stacking	BLR-stacking	AdaBoost-stacking	SVR-stacking	CART-stacking	ELM-stacking
Validation dataset	Stage A	4.892	5.247	4.181	4.714	3.987	4.208
	Stage B	4.513	4.506	4.707	4.681	4.082	4.405
	Average	4.703	4.877	4.444	4.698	4.035	4.307
Test dataset	Stage A	5.863	6.521	5.367	5.209	5.065	4.795
	Stage B	4.794	4.667	4.548	4.579	4.500	4.485
	Average	5.329	5.594	4.958	4.894	4.783	4.640
Regressor training time		0.783	44.734	38.943	84.763	0.002	0.580

Appendix: Description of the 35 Features Used in This Study ($F = [F_1, F_2, \ldots, F_{35}]$)

Feature	Description	Feature	Description	Feature	Description
F_1	Central moment of x_7	F_{13}	Skewness of x_{14}	F_{25}	Standard deviation of x_{14}
F_2	Central moment of x_9	F_{14}	Skewness of x_{15}	F_{26}	Standard deviation of x_{16}
F_3	Central moment of x_{10}	F_{15}	Skewness of x_{16}	F_{27}	Standard deviation of x_{19}
F_4	Central moment of x_{15}	F_{16}	Skewness of x_{19}	F_{28}	Standard deviation of x_{21}
F_5	Central moment of x_{17}	F_{17}	Skewness of x_{20}	F_{29}	Standard deviation of x_{25}
F_6	Central moment of x_{18}	F_{18}	Skewness of x_{25}	F_{30}	Kurtosis of x_7
F_7	Central moment of x_{23}	F_{19}	Standard deviation of x_7	F_{31}	Kurtosis of x_{12}
F_8	Central moment of x_{25}	F_{20}	Standard deviation of x_8	F_{32}	Kurtosis of x_{22}
F_9	Skewness of x_7	F_{21}	Standard deviation of x_9	F_{33}	Kurtosis of x_{23}
F_{10}	Skewness of x_8	F_{22}	Standard deviation of x_{10}	F_{34}	Kurtosis of x_{24}
F_{11}	Skewness of x_{10}	F_{23}	Standard deviation of x_{11}	F_{35}	Kurtosis of x_{25}
F_{12}	Skewness of x_{12}	F_{24}	Standard deviation of x_{12}		

References

[1] Krishnan, M., Nalaskowski, J. W., and Cook, L. M., 2009, "Chemical Mechanical Planarization: Slurry Chemistry, Materials, and Mechanisms," Chem. Rev., **110**(1), pp. 178–204.

[2] Steigerwald, J. M., Murarka, S. P., and Gutmann, R. J., 2008, *Chemical Mechanical Planarization of Microelectronic Materials*, Wiley, New York.

[3] Nanz, G., and Camilletti, L. E., 1995, "Modeling of Chemical-Mechanical Polishing: A Review," IEEE Trans. Semicond. Manuf., **8**(4), pp. 382–389.

[4] Evans, C., Paul, E., Dornfeld, D., Lucca, D., Byrne, G., Tricard, M., Klocke, F., Dambon, O., and Mullany, B., 2003, "Material Removal Mechanisms in Lapping and Polishing," CIRP Ann.-Manuf. Technol., **52**(2), pp. 611–633.

[5] Luo, Q., Ramarajan, S., and Babu, S., 1998, "Modification of the Preston Equation for the Chemical–Mechanical Polishing of Copper," Thin Solid Films, **335**(1–2), pp. 160–167.

[6] Luo, J., and Dornfeld, D. A., 2001, "Material Removal Mechanism in Chemical Mechanical Polishing: Theory and Modeling," IEEE Trans. Semicond. Manuf., **14**(2), pp. 112–133.

[7] Yu, T., Asplund, D. T., Bastawros, A. F., and Chandra, A., 2016, "Performance and Modeling of Paired Polishing Process," Int. J. Mach. Tools Manuf., **109**, pp. 49–57.

[8] Kong, Z., Oztekin, A., Beyca, O. F., Phatak, U., Bukkapatnam, S. T., and Komanduri, R., 2010, "Process Performance Prediction for Chemical Mechanical Planarization (CMP) by Integration of Nonlinear Bayesian Analysis and Statistical Modeling," IEEE Trans. Semicond. Manuf., **23**(2), pp. 316–327.

[9] Rao, P. K., Beyca, O. F., Kong, Z., Bukkapatnam, S. T., Case, K. E., and Komanduri, R., 2015, "A Graph-Theoretic Approach for Quantification of Surface Morphology Variation and Its Application to Chemical Mechanical Planarization Process," IIE Trans., **47**(10), pp. 1088–1111.

[10] Wang, J., Ma, Y., Zhang, L., Gao, R. X., and Wu, D., 2018, "Deep Learning for Smart Manufacturing: Methods and Applications," J. Manuf. Syst., **48**(C), pp. 144–156.

[11] Wu, D., Jennings, C., Terpenny, J., Gao, R. X., and Kumara, S., 2017, "A Comparative Study on Machine Learning Algorithms for Smart Manufacturing: Tool Wear Prediction Using Random Forests," ASME J. Manuf. Sci. Eng., **139**(7), p. 071018.

[12] Wu, D., Jennings, C., Terpenny, J., Kumara, S., and Gao, R. X., 2018, "Cloud-Based Parallel Machine Learning for Tool Wear Prediction," ASME J. Manuf. Sci. Eng., **140**(4), p. 041005.

[13] Lin, S.-C., and Wu, M.-L., 2002, "A Study of the Effects of Polishing Parameters on Material Removal Rate and Non-Uniformity," Int. J. Mach. Tools Manuf., **42**(1), pp. 99–103.

[14] Lee, H., and Jeong, H., 2011, "A Wafer-Scale Material Removal Rate Profile Model for Copper Chemical Mechanical Planarization," Int. J. Mach. Tools Manuf., **51**(5), pp. 395–403.

[15] Lee, H., Jeong, H., and Dornfeld, D., 2013, "Semi-Empirical Material Removal Rate Distribution Model for SiO_2 Chemical Mechanical Polishing (CMP) Processes," Precis. Eng., **37**(2), pp. 483–490.

[16] Lih, W.-C., Bukkapatnam, S. T., Rao, P., Chandrasekharan, N., and Komanduri, R., 2008, "Adaptive Neuro-Fuzzy Inference System Modeling of MRR and WIWNU in CMP Process With Sparse Experimental Data," IEEE Trans. Autom. Sci. Eng., **5**(1), pp. 71–83.

[17] Wang, P., Gao, R. X., and Yan, R., 2017, "A Deep Learning-Based Approach to Material Removal Rate Prediction in Polishing," CIRP Ann., **66**(1), pp. 429–432.

[18] Jia, X., Di, Y., Feng, J., Yang, Q., Dai, H., and Lee, J., 2018, "Adaptive Virtual Metrology for Semiconductor Chemical Mechanical Planarization Process Using GMDH-Type Polynomial Neural Networks," J. Process Control, **62**, pp. 44–54.

[19] Rao, P. K., Bhushan, M. B., Bukkapatnam, S. T., Kong, Z., Byalal, S., Beyca, O. F., Fields, A., and Komanduri, R., 2014, "Process-Machine Interaction (PMI) Modeling and Monitoring of Chemical Mechanical Planarization (CMP) Process Using Wireless Vibration Sensors," IEEE Trans. Semicond. Manuf., **27**(1), pp. 1–15.

[20] Džeroski, S., and Ženko, B., 2004, "Is Combining Classifiers With Stacking Better Than Selecting the Best One?," Mach. Learn., **54**(3), pp. 255–273.

[21] Zhou, Z.-H., 2012, *Ensemble Methods: Foundations and Algorithms*, Chapman & Hall, Boca Raton, FL.

[22] Friedman, J. H., 2001, "Greedy Function Approximation: A Gradient Boosting Machine," Ann. Stat., **29**(5), pp. 1189–1232.

[23] Li, Z., Wu, D., Hu, C., and Terpenny, J., 2017, "An Ensemble Learning-Based Prognostic Approach With Degradation-Dependent Weights for Remaining Useful Life Prediction," Reliab. Eng. Syst. Saf., (in Press).

[24] Geurts, P., Ernst, D., and Wehenkel, L., 2006, "Extremely Randomized Trees," Mach. Learn., **63**(1), pp. 3–42.

[25] Breiman, L., 2001, "Random Forests," Mach. Learn., **45**(1), pp. 5–32.

[26] Liaw, A., and Wiener, M., 2002, "Classification and Regression by random Forest," R News, **2**(3), pp. 18–22.

[27] Ho, T. K., 1998, "The Random Subspace Method for Constructing Decision Forests," IEEE Trans. Pattern Anal. Mach. Intell., **20**(8), pp. 832–844.

[28] Friedman, J. H., 2002, "Stochastic Gradient Boosting," Comput. Stat. Data Anal., **38**(4), pp. 367–378.

[29] Rosca, N. P. J., 2016, "PHM Society Data Challenge," PHM Society, Denver, CO, accessed Nov. 30, 2018, https://www.phmsociety.org/events/conference/phm/16/data-challenge

[30] Ki Bum, L., and Ouk Kim, C., 2018, "Recurrent Feature-Incorporated Convolutional Neural Network for Virtual Metrology of the Chemical Mechanical Planarization Process," J. Intell. Manuf., pp. 1–14.

[31] Greenwood, J., and Williamson, J. P., 1966, "Contact of Nominally Flat Surfaces," Proc. R. Soc. London, A, **295**(1442), pp. 300–319.

[32] Johnson, K. L., 1987, *Contact Mechanics*, Cambridge University Press, Cambridge, UK.

[33] Seber, G. A., and Lee, A. J., 2012, *Linear Regression Analysis*, Wiley, Hoboken, NJ.

[34] Makalic, E., and Schmidt, D. F., 2016, "High-Dimensional Bayesian Regularised Regression With the BayesReg Package," preprint arXiv:1611.06649.

[35] Kang, P., Kim, D., and Cho, S., 2016, "Semi-Supervised Support Vector Regression Based on Self-Training With Label Uncertainty: An Application to Virtual Metrology in Semiconductor Manufacturing," Expert Syst. Appl., **51**, pp. 85–106.

[36] Solomatine, D. P., and Shrestha, D. L., 2004, "AdaBoost.RT: A Boosting Algorithm for Regression Problems," IEEE International Joint Conference on Neural Networks, Budapest, Hungary, July 23–29, pp. 1163–1168.