IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 28, 2020, accepted March 21, 2020, date of publication March 25, 2020, date of current version April 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2983188

Prediction for Manufacturing Factors in a Steel Plate Rolling Smart Factory Using Data Clustering-Based Machine Learning

CHEOL YOUNG PARK[1], (Member, IEEE), JIN WOOG KIM[2], BOSUNG KIM[3], AND JOONGYOON LEE[4]

[1]Bayesian AI Laboratory, BAIES, Fairfax, VA 22030, USA
[2]Deep Learning Laboratory, DEEP-IN, Seoul 6079, South Korea
[3]POSCO, Pohang 37822, South Korea
[4]GIFT, POSTECH University, Pohang 37673, South Korea

Corresponding author: Joongyoon Lee (jlee2012@postech.ac.kr)

This work was supported in part by the POSCO under Grant 2019Y048.

ABSTRACT A Steel Plate Rolling Mill (SPM) is a milling machine that uses rollers to press hot slab inputs to produce ferrous or non-ferrous metal plates. To produce high-quality steel plates, it is important to precisely detect and sense values of manufacturing factors including plate thickness and roll force in each rolling pass. For example, the estimation or prediction of the in-process thickness is utilized to select the control values (e.g., roll gap) in the next pass of rolling. However, adverse manufacturing conditions can interfere with accurate detection for such manufacturing factors. Although the state-of-the-art gamma-ray camera can be used for measuring the thickness, the outputs from it are influenced by adverse manufacturing conditions such as the high temperature of plates, followed by the evaporation of lubricant water. Thus, it is inevitable that there is noise in the thickness estimation. Furthermore, installing such thickness measurements for each passing step is costly. The precision of the thickness estimation, therefore, significantly affects the cost and quality of the final product. In this paper, we present machine learning (ML) technologies and models that can be used to predict the in-process thickness in the SPM operation, so that the measurement cost for the in-process thickness can be significantly reduced and high-quality steel plate production can be possible. To do so, we investigate most-known technologies in this application. In particular, Data Clustering based Machine Learning (DC-ML), combining clustering algorithms and supervised learning algorithms, is introduced. To evaluate DC-ML, two experiments are conducted and show that DC-ML is well suited to the prediction problems in the SPM operation. In addition, the source code of DC-ML is provided for the future study of machine learning researchers.

INDEX TERMS Intelligent manufacturing systems, machine learning, regression analysis, steel industry, thickness control.

I. INTRODUCTION

As the fourth industrial revolution, called *Industry 4.0*, becomes more pervasive, contemporary manufacturing also becomes smarter using state-of-the-art technologies such as artificial intelligence, cloud computing, internet of things, cyber-physical systems, and big data. These technologies make smart manufacturing [1]–[12] radically feasible. In this paper, we introduce an application of ML technologies in a steel plate production smart factory.

In a steel plate factory line, the input of the line is a slab made by continuous casting of molten steel and the output of the line is a steel plate. And the steel plate is produced by a special facility, a *Steel Plate Rolling Mill* (SPM). The rolling process is a metal forming process in which a slab is passed through a set of rolls in order to uniformly reduce the thickness of the slab by handling the gap of rolls. To produce high-quality steel plates, it is important to precisely detect and sense values of manufacturing factors such as roll gap, roll force, and temperature. However, environmental factors such as high temperature can hinder accurate value detection for manufacturing factors (e.g., the thickness of a steel plate when passing through the SPM). In a steel plate factory

The associate editor coordinating the review of this manuscript and approving it for publication was Berdakh Abibullaev.

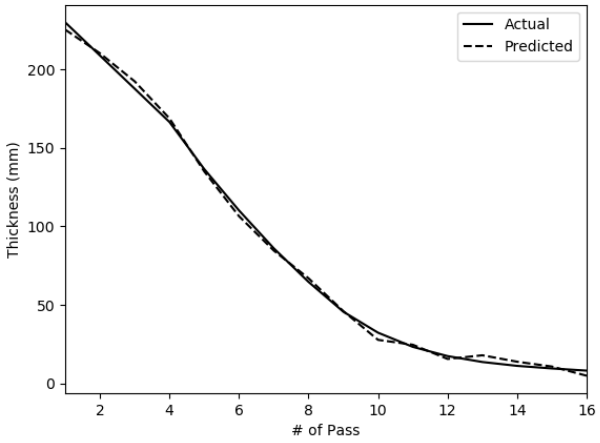


FIGURE 1. Actual thickness vs predicted thickness.

line, the estimation of the in-process thickness is utilized to select the control values (e.g., roll gap) for the next pass. The precision of the thickness estimation, therefore, can significantly affect the final product. Although a gamma-ray camera can be used, the outputs from it can be influenced by adverse manufacturing conditions such as high temperature of plates and followed evaporation of lubricant water. Thus, it is inevitable that there is noise in the thickness estimation using such thickness measuring sensors. Furthermore, the use of such sensors causes another production cost.

In this paper, we introduce machine learning approaches predicting manufacturing factors to support existing SPM control systems. Figure 1 shows one illustrative example of the problem in this paper. Over the number of rolling passes (x-axis), the SPM control systems require the high precision for value estimation of manufacturing factors to produce the required thickness (y-axis) of the plate in the next rolling step. To do so, the measured manufacturing data from sensors are used to predict the next thickness of the plate. For high-quality production, the prediction for the next thickness (the dashed line) should be as close as possible to the actual value (the solid line).

Specifically, this paper introduces four existing machine learning approaches and one novel machine learning algorithm in order to support the SPM control systems. One of traditional SPM control systems is *Automatic Gauge Control* (AGC) in steel plate production. AGC has been successfully applied to commercial rolling mill system to select required control values. Usually, the conventional AGC systems are based on *Proportional Integral Derivative* (PID) controller [13]–[15], a feedback-loop mechanism adjusting control values to address target values. Such PID controllers are widely used in industrial control systems (e.g., temperature control, flow control, pneumatic control, and compressor control), including AGC systems. For example, Zhang *et al.* [16] introduced a generalized predictive control algorithm, evolved from the existing control algorithms for hydraulic AGC. They used a simulation for evaluation and showed an improved thickness precision of strips. Karandaev *et al.* [17] applied a transfer function to an AGC control model in order to address the control error of the existing AGC, so that they could reduce gauge deviations. Zhang and Ding [18] introduced a strategy of the AGC control to improve the final product quality. The control limitations of the conventional AGC control under compound disturbance were addressed by using such a control strategy that could remove rolling uncertainty in the AGC operation. However, to develop a PID-based AGC, the mathematical models are required and designed by subject-matter experts (SME). Furthermore, designing such mathematical models is not practicable, when considering a lot of manufacturing factors. Consequently, simple PID models have been developed. Another drawback of PID controllers is that it is not easy to deal with the complex non-linearity [19]. To overcome these limitations of the conventional AGC controllers, self-adjusting AGC systems, developing control models automatically, have been researched and developed. For example, *Fuzzy Logic* [20] based control systems were presented. The fuzzy control systems have several advantages such as human understandable model, fast and easy implementation, ability to deal with non-linearity, and so on. Wang *et al.* [21] utilized a fuzzy control system to perform self-adjusted PID controller in an AGC system. The stimulation results of the paper showed that the proposed fuzzy system outperformed conventional PID systems. However, because such fuzzy systems are based on various domain assumptions and human interventions, the reasoning results can be inaccurate. In addition, it is not trivial to design fuzzy rules by SME (i.e., dependent on the domain knowledge level).

As another example of the self-adjusting AGC systems, some researchers have focused on the prediction of a roll force value. One critical factor of designing a conventional control model of the AGC systems is the roll force. Selecting a precise roll force value for each rolling process affects the quality of thickness reduction of a steel plate [22]. In this research domain, *Artificial Neural Networks* (ANN) were used to predict the roll force value [23]–[29]. Lee and Choi [23] applied ANN to roll force prediction. Their results showed the 30% improvement of the final product quality. Zhang *et al.* [24] combined differential evolution with ANN. The prediction error of the proposed approach was less than 5%. Rath *et al.* [25] applied ANN for prediction of roll force. They used a feed forward network as an ANN architecture and a back propagation algorithm. A conjugate gradient optimization of the loss function is used for network training. The prediction accuracy of the trained model was the R-squared value of about 0.94. Bagheripoor and Bisadi [26] applied ANN and used the similar feed forward network and back propagation algorithm. The prediction accuracy of the trained model was the R-squared value of about 0.979. Wang *et al.* [27] used an ANN for the bending force prediction in a hot strip rolling. They suggested the ANN architecture which was optimized by a genetic algorithm and Bayesian regulation. The prediction accuracy of the proposed

architecture was the R-squared value of 0.956. Liu *et al.* [28] applied a genetic algorithm (GA), particle swarm optimization algorithm (PSO), and multiple hidden layer extreme learning machine (MELM) for their model. They used GA to determine the optimal number of hidden layers and the optimal number of hidden nodes. PSO was used to search for the optimal input weights and biases. Esendağ *et al.* [29] used ANN and conventional regression models (e.g., Support Vector Machines) to predict reversible cold rolling process parameters. They reported the roll force prediction accuracy for the ANN as the R-squared value of 0.939 and the regression model as the R-squared value of 0.947.

In this paper, several ML algorithms are used to predict two main parameters of the SPM control systems. One is the roll force, because the plate thickness can be directly calculated by using the roll force value. Another is the plate thickness at each rolling pass, so that we can find the best control conditions without an expensive sensor (e.g., the gamma-ray camera) and its operational cost. Furthermore, high-quality plate production of the SPM control systems can be achieved. Specifically, four well-known ML regression models are utilized for these two predictions.

(1) Random Forest Regression (RF)
(2) Gradient Boosting Regression (GB)
(3) Gaussian Process Regression (GP)
(4) Conditional Linear Gaussian (CG)

In addition, this paper introduces *Data Clustering based Machine Learning* (DC-ML). DC-ML is based on an idea in which training data for machine learning are classified into a set of data by clustering and then each data of the set are learned by supervised learning, including regression and classification.

There are similar studies regarding data clustering based machine learning. Wang *et al.* [30] introduced clustering-based *Kriging*, or Gaussian Process Regression [31], to solve the problem of *Efficient Global Optimization* (EGO). Kriging has the advantage of learning a complex function. However, when it is required to be processed with large data, a problem arises in computing large matrix multiplication. To solve such a big data problem, the paper introduced how to combine a clustering algorithm with Kriging for EGO. Qiang *et al.* [32] presented an algorithm regarding a clustering-based artificial neural network. In the initial step of the algorithm, many neural networks are trained. And then these networks are divided into clusters using K-means clustering [33] according to the output results of each network. The most accurate one network in each cluster is selected to be used for inference.

These previous studies focused on the specific clustering and classification algorithm (i.e., Kriging and artificial neural network). In this paper, we introduce a general algorithm in terms of data clustering based machine learning. The presented algorithm utilizes existing clustering and supervised learning algorithms to make a group of clustering and supervised learning models. For a performance analysis, two experiments are conducted and show that the presented DC-ML is well suited to the prediction problems in the SPM control systems and outperforms the above four regression models (RF, GB, GP, and CG).

This paper contributes to three research agendas: (1) suggest DC-ML in the application of SPM, (2) provide the source code for DC-ML, and (3) introduce the experiment results using the real-world data from a steel plate rolling smart factory.

The remainder of the paper is organized as follows. Section 2 introduces background knowledge on the concept of SPM, the basic theory of thickness reduction of SPM, and the machine learning algorithms used in this paper. Section 3 suggests the algorithm of DC-ML. Section 4 presents the experiments regarding roll force and plate thickness prediction in SPM. Section 5 discusses the experiment results in terms of prediction accuracy. The final section presents conclusions and future research directions.

II. BACKGROUND

In this section, we introduce the concept of the rolling mill process, the basic theory of thickness reduction, and machine learning technologies regarding regression. This prerequisite knowledge will be the basis of the methodology introduced in Section 3.

A. RESEARCH TARGET SYSTEM

The steel plate factory process usually contains seven steps to produce a steel plate using a slab: (1) Reheating Furnace, (2) Hot Scale Breaker, (3) Input Size Measure, (4) Rolling Mill Stand, (5) Output Size Measure, (6) Cooling, and (7) Hot Leveler. The steel plate smart factory in this paper has only one rolling mill stand, which performs multiple reciprocating pass operations to enlarge the width and/or length of the steel plate, and reduce the thickness of it to achieve the desired target size. In this paper, the target rolling mill system is a four-high reciprocating rolling mill stand. The specification of this machine includes 8,000 tons of rolling capacity, 4 meters of rolling width, and 5 m/sec of rolling speed. It is equipped with the pair-cross automatic gauge control system.

B. BASIC THEORY OF THICKNESS REDUCTION BY ROLLING MILL OPERATION

The rolling process is a metal forming process in which a slab is passed through a set of rolls to uniformly reduce the thickness of the slab by handling the gap of rolls. Equation 1 represents the relation of the output thickness Th and the roll gap SD under ideal conditions.

$$Th(i+1) = SD, \tag{1}$$

where SD denotes *Screw Down* of mill (simply, Roll Gap) and i denotes the rolling pass number.

That is, when a thick plate $Th(i)$ is input to the roll from the left side (Figure 2), the plate with reduced thickness $Th(i+1)$ by the roll gap SD is output to the right side.

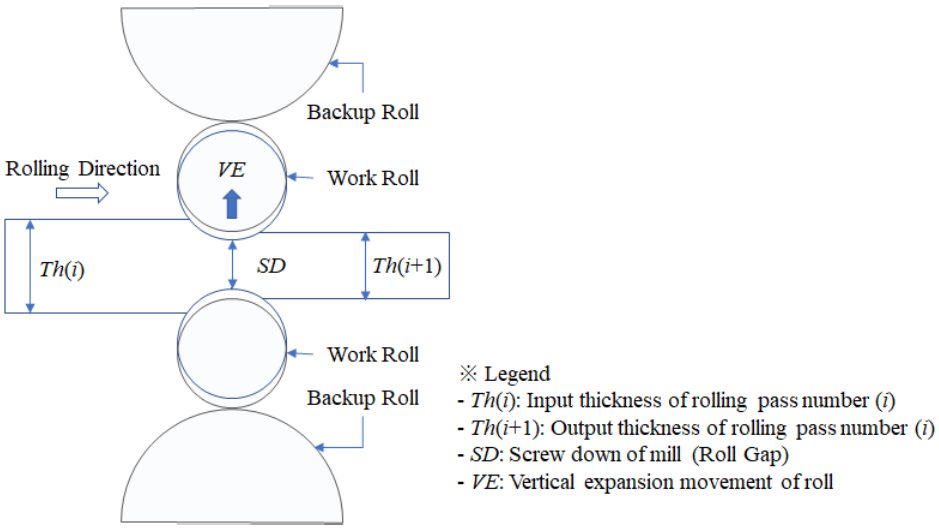


FIGURE 2. Thickness reduction concept by rolling.

1) PROBLEM OF THICKNESS CONTROL IN ROLLING

The concept of the rolling process is simple, however precise thickness control is not trivial, because of the various noise factors like vertical expansion movement of roll, shape deformation (roll crown), temperature interaction factor, and so on. The following introduces some explanation of major noise factors which should be considered in the rolling process.

- **Vertical Expansion Movement of Roll**

  As shown in Figure 2, the vertical expansion (VE) movement of roll occurs due to the material repulsion force (Roll Force) for the thickness reduction in the rolling process. The vertical expansion should be reflected when setting the roll gap in order to meet the target thickness of the output plate. The value of VE can be obtained by dividing the value of roll force by Mill Modulus (MM). And also, the value of roll force can be obtained by the high temperature strength, thickness reduction rate, width of the rolling plate, and rolling speed. In addition, when setting the roll gap, it should be considered that the value of MM is slightly changed by the width of the plate.

- **Shape Deformation of Roll**

  The original convex cylinder form of the roll (roll crown) can be flattened due to abrasion, as rolling quantities increase. Such a shape deformation also should be reflected, when setting the roll gap.

- **Rolling Temperature**

  Under the rolling mill operation, the temperature directly contacted with the plate can rise and the roll can be expanded due to the heat of the rolling plate. And the plate is shrunk due to cool down from high rolling temperature. This thermal expansion of the roll and cooling shrink of the plate should be reflected, when setting the roll gap.

- **Plate Dimension**

  During rolling the input plate, the thickness and width especially vary for each rolling pass. The difference of such dimensions causes the different mill modulus and roll force, and eventually leads to a different roll gap.

- **Other Noise Factors**

  In addition to the above major noise factors, Roberts [34] introduced more factors like the coefficient of friction, work-roll diameter, and rolling speed related to the mathematical models for predicting the roll force. Such factors associated with the roll force are also related to the thickness of the plate. Gingzburg and Ballas [35] suggested that the disturbances, affecting gauge performance in rolling mills, can be caused by various sources. Table 1 summarizes these noise factors.

2) EQUATION FOR THE OUTPUT THICKNESS

In the previous subsection, the basic Equation 1 was only associated with the roll gap SD. Equation 2 represents the relationship between the output thickness and roll gap under the various noise conditions [36], [37] [38].

$$Th(i+1) = (SD + \frac{RLF}{MM} - S) \times TF, \tag{2}$$

where RLF denotes the roll force, MM denotes the mill modulus which includes compensation for the plate width variation, S denotes the adjustment of the roll gap which includes compensation for thickness variation, strength variation and others, and TF denotes the thermal shrinkage compensation factor. Details can be found in [36], [37] [38]. Equation 2 is

TABLE 1. Main factors affecting gauge performance in rolling mills [35].

Source of Disturbances	Factor Groups	Factors
Disturbances from mill mechanical and hydraulic equipment	No-load roll gap	Roll Bearing Oil Film Thickness, Roll Ovality, Mill Chatter, Roll Balance Force, Roll Bite Lubricant Film, Thickness, Roll Expansion or Contraction, Roll Wear, and Roll Eccentricity
Disturbances from mill mechanical and hydraulic equipment	Main factors affecting mill stiffness	Roll flattening, Roll Crown, Hydraulic Cylinder Extension, Roll Bit Lubricant Film Thickness, Rolled Material Width, Bearing Oil Film Thickness, Screw Down Extension, Roll and Diameter
Disturbances from mill control systems	Mill control systems affecting gauge performance	Mill Speed Control, Roll Force Control, Roll Balance Control, Strip Tension Control, Gauge Monitor Control, Roll Coolant and Lubrication Control, Roll Bending Control, and Roll Gap Control
Disturbance from incoming rolled product	Geometry variations of incoming product	Gauge Variation, Hardness Variation, Width Variation, Profile Variation, and Flatness Variation

used for the basis of developing a causal model introduced in Subsection 4.B.

C. REGRESSION IN MACHINE LEARNING

A *Regression* model is used to predict a continuous response $f(X)$, or a target variable, using predictor variables $X = \{x_1, x_2, \ldots, x_n\}$. This paper uses four well-known regression models: (1) *Random Forest Regression*, (2) *Gradient Boosting Regression*, (3) *Gaussian Process Regression*, and (4) *Conditional Linear Gaussian*. Random Forest Regression can handle large data, missing data, and many variables. However, for unseen data, it can not predict a continuous change precisely. Also, it can be over-fitted for noisy data and the learned model from Random Forest Regression is difficult to be interpreted. Gradient Boosting Regression is prone to over-fitting, so it requires careful hyperparameter tuning, when performing machine learning. Gaussian Process Regression is a promising model in regression. It can predict continuous change in nonlinear regression. However, it is not suitable for large data [39]. Conditional Linear Gaussian is a simple and human editable model that allows subject-matter experts to modify it. However, it is a linear model. In this subsection, these four models are briefly introduced.

1) RANDOM FOREST REGRESSION

A set of ML models can often have a better performance than the use of a single ML model. Such an integration of ML models is called ensemble learning. *Random Forest* [40] uses the ensemble learning by forming a set of decision trees (e.g., Classification and Regression Tree, CART [41]) and resulting in an output which is averaged over outputs from the decision trees. Random Forest draws random samples from training data and creates a decision tree model from the sample data, so that it can have a set of decision trees (i.e., forest). After machine learning, in the prediction or application stage, the mean value of the outputs of all decision trees is yielded as the final result. Equation 3 shows an equation for the averaging outputs from the set of the learned decision trees.

$$\hat{y} = Mean\{a_1(x), a_2(x), \ldots, a_n(x)\}, \tag{3}$$

where $a_i(x)$ is a single decision tree and the function $Mean(.)$ yields the average value using the outputs from the set of the decision trees.

2) GRADIENT BOOSTING REGRESSION

Gradient Boosting [42] uses an ensemble model consisting of a set of simple models (e.g., a decision tree stump, a tree containing only one root and its immediately connected leaf nodes). By adding such simple models, the result ensemble model can be sequentially improved and finally fitted to data. In other words, after applying a simple model, samples which are classified by it are reused to another simple model. And then this process is repeated until convergence (or achieving better predictive performance). Gradient Boosting is a generalized method of boosting (e.g., [43], [44]) by using gradient of a loss function.

3) GAUSSIAN PROCESS REGRESSION

Gaussian Process is composed of a set of Gaussian random variables, specified by a mean and covariance (or kernel) function. Equation 4 formally shows Gaussian Process [45].

$$P(F(x)|D, x) = N(\mu(x),\ \sigma^2(x)), \tag{4}$$

where D denotes an observed data $\{x_{1:n}, F(x_{1:n})\}$, x denotes an independent value for $F(.)$, $N(.,.)$ denotes a normal distribution, $\mu(.)$ denotes a mean function of x, and $\sigma^2(.)$ denotes

a variance function of x. These $\mu(.)$ and $\sigma^2(.)$ are shown in Equations 5 and 6, respectively.

$$\mu(x) = \mathbf{k}^T \mathbf{K}^{-1} F(x_{1:n}) \tag{5}$$

and

$$\sigma^2(x) = \mathbf{k}(x, x) - \mathbf{k}^T \mathbf{K}^{-1} \mathbf{k}, \tag{6}$$

where $\mathbf{k} = [k(x, x_1), k(x, x_2), \ldots, k(x, x_n)]$ denotes a set of kernel functions $k(., .)$ and $\mathbf{K}$ denotes a kernel matrix as shown by Equation 7.

$$\mathbf{K} = \begin{bmatrix} k(x_1, x_1) & \ldots & k(x_1, x_n) \\ \ldots & \ldots & \ldots \\ k(x_n, x_1) & \ldots & k(x_n, x_n) \end{bmatrix} \tag{7}$$

Using Equations 5 and 6, it is straightforward to compute *Gaussian Process* in Equation 4.

4) CONDITIONAL LINEAR GAUSSIAN BAYESIAN NETWORK FOR REGRESSION

Conditional Linear Gaussian (CLG) *Bayesian Network* (BN) (CLG-BN) [46] can be used for the regression problem in this paper. Also, CLG-BN can be used to estimate the posterior probability distribution for the target variable using various reasoning algorithms [47]–[49]. Parameters in conditional linear Gaussian distribution can be estimated by using an extension of multiple-regression.

In CLG-BN, we assume that X is a continuous node with n continuous parents $\mathrm{U}_1, \ldots, \mathrm{U}_\mathrm{n}$ and m discrete parents $\mathrm{A}_1, \ldots, \mathrm{A}_\mathrm{m}$, then the conditional distribution p(X | **u**, **a**) given parent states $\mathbf{U} = \mathbf{u}$ and $\mathbf{A} = \mathbf{a}$ has the following form:

$$p(X|\mathbf{u}, \mathbf{a}) = N(L^{(\mathrm{a})}(\mathbf{u}), \sigma^{(\mathrm{a})}), \tag{8}$$

where $L^{(\mathrm{a})}(\mathbf{u}) = \mathrm{m}^{(\mathrm{a})} + \mathrm{b}_1{}^{(\mathrm{a})}\mathrm{u}_1 + \ldots + \mathrm{b}_\mathrm{n}{}^{(\mathrm{a})}\mathrm{u}_\mathrm{n}$ is a linear function of the continuous parents, with intercept $\mathrm{m}^{(\mathrm{a})}$, coefficients $\mathrm{b}_\mathrm{i}{}^{(\mathrm{a})}$, and standard deviation $\sigma^{(\mathrm{a})}$ that depends on the state a of the discrete parents.

Given a discrete parent state a_j, estimating the parameters (i.e., the intercept $\mathrm{m}^{(\mathrm{a_j})}$, coefficients $\mathrm{b}_\mathrm{i}{}^{(\mathrm{a_j})}$, and standard deviation $\sigma^{(\mathrm{a_j})}$) is required. Equation 9 shows multiple linear regression which is modified from [50]. $L^{(\mathrm{a})}(\mathbf{u})$ can be rewritten, if we suppose that there are k observations (or data) (Note that in the following, we can omit the state **a**, because we know it).

$$L_\mathrm{i}(\mathbf{u}) = m + b_1 u_{\mathrm{i}1} + \ldots + b_\mathrm{n} u_{\mathrm{in}} + \sigma_\mathrm{i}, i = 1, \ldots, k, \tag{9}$$

where i indexes the observations. For convenience, we can write Equation 9 more compactly using matrix notation:

$$\mathbf{l} = \mathbf{U}\mathbf{b} + \boldsymbol{\sigma}, \tag{10}$$

where **l** denotes a vector of instances for the observations, **U** denotes a matrix containing all continuous parents in the observations, **b** denotes a vector containing an intercept m and a set of coefficients $\boldsymbol{b_i}$, and $\boldsymbol{\sigma}$ denotes a vector of regression residuals. Equation 11 show these variables in forms of vectors and a matrix.

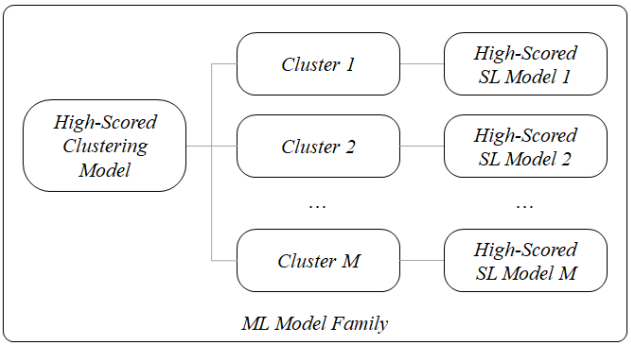


FIGURE 3. A machine learning model family.

$$\mathbf{l} = \begin{bmatrix} L_1(\boldsymbol{u}) \\ L_1(\boldsymbol{u}) \\ \ldots \\ L_1(\boldsymbol{u}) \end{bmatrix} \mathbf{U} = \begin{bmatrix} 1 & u_{11} & \ldots & u_{1\mathrm{n}} \\ 1 & u_{21} & \ldots & u_{2\mathrm{n}} \\ \ldots & \ldots & \ldots & \ldots \\ 1 & u_{\mathrm{k}1} & \ldots & u_{\mathrm{kn}} \end{bmatrix} \mathbf{b} = \begin{bmatrix} m \\ b_1 \\ \ldots \\ b_\mathrm{k} \end{bmatrix} \boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \ldots \\ \sigma_\mathrm{k} \end{bmatrix} \tag{11}$$

From the above settings, we can derive an optimal vector for the intercept and the set of coefficients $\hat{\mathbf{b}}$

$$\hat{\mathbf{b}} = (\mathbf{U}^\mathrm{T}\mathbf{U})^{-1}\mathbf{U}^\mathrm{T}\mathbf{l}, \tag{12}$$

Also, we can derive the optimal standard deviation $\hat{\sigma}$ from the above linear algebra term [50].

$$\hat{\sigma} = \sqrt{\frac{(\mathbf{l} - \mathbf{U}\hat{\mathbf{b}})^\mathrm{T}(\mathbf{l} - \mathbf{U}\hat{\mathbf{b}})}{k - n - 1}} \tag{13}$$

In summary, using observation (or data) **U**, Equation 12, and Equation 13, we can simply form Equation 10 and Equation 9. In this paper, we used a probabilistic graphical modeling package, called UnBBayes [51], which contains a CLG-BN machine learning algorithm [52].

III. DATA CLUSTERING BASED MACHINE LEARNING

In this section, we introduce *Data Clustering based Machine Learning* (DC-ML). In supervised learning, the training data consist of data for predictor variables (e.g., X variables) and data for a target variable (e.g., a Y variable). The data for the predictor variables may or may not be classified as several clusters. If the data clusters exist, we can imply that there are several corresponding forces promoting such clusters. These forces may differently influence the target variable. If that is the case, separating data according to the clusters would be better than using all data for supervised learning. In this case, each clustered data is used to learn a corresponding supervised learning model. Consequently, a *machine learning model family* or *ML model family*, containing a set of ML models, is constructed (Figure 3).

Figure 3 shows an illustrative example of an ML model family. The ML model family contains a high-scored clustering model consisting of M clusters. The high-scored clustering or supervised learning model used herein refers to

the model which is selected by the highest score against other candidate models. The score (e.g., R-Squared Score and Mean Absolute Error) can be determined by an analysis goal. Each cluster is associated with a corresponding high-scored supervised learning model (a regression model or classification model). Such an ML model family is learned by DC-ML as shown by Figure III. Figure III illustrates three main functions: (1) *Perform DC-ML*, (2) *Perform Clustering (CL) and Split Data according to Clusters*, and (3) *Perform Supervised Learning (SL)*. The first step of DC-ML starts with training data. Several clustering algorithms are independently used to data clustering and generate clustering models from 1 to L. Each clustering model contains clusters by which the training data are split into clustered data from 1 to M. Each clustered data are used to perform supervised learning. By supervised learning, SL models from 1 to N are output. Several ML model families containing clustering models and SL models are generated by this process and then one high-scored ML model family is selected as the output of DC-ML. After the high-scored ML model family is learned by DC-ML, it can be used for prediction. Figure 5 illustrates two main functions of the DC-ML prediction: (1) *Select an SL Model according to Data Clusters* and (2) *Perform Prediction*. The first step of prediction starts with data. The clustering model in the ML model family is utilized to select an SL model using the given data. The given data are reused to predict a target value by using the selected SL model.

In Algorithm 1, DC-ML is described in more detail. DC-ML has five inputs. The first input $\boldsymbol{D}_X$ is the training data set for predictor variables. The second input $\boldsymbol{D}_Y$ is the training data set for a target variable. The third input $\boldsymbol{C}$ is the set of clustering algorithms (e.g., *Gaussian Mixture* [53], *Birch* [54], and *Mini Batch K-Means* [33]). The input of each clustering algorithm contains a set of candidate hyperparameters (e.g., *Gaussian Mixture* algorithms associated with 2, 3, 4, and 5 clusters, respectively). The fourth input $\boldsymbol{S}$ is the set of supervised learning algorithms (e.g., *Random Forest Regression*, *Gradient Boosting Regression*, and *Gaussian Process Regression*). The supervised learning algorithm can also take the candidate hyperparameters. The fifth input $\boldsymbol{V}$ is the set of *clustering variables*. For clustering, it is not necessary that all the variables for the training data are used. The clustering variables means the variables that are selected to be used for clustering. Given these inputs, Algorithm 1 proceeds as follows:

Line 1 The algorithm starts with the function *Run*(.).

Line 2 The function *Run*(.) iterates the function *Perform Clustering*(.) in parallel. To do that, an index i is taken from 1 to the number of clustering algorithms in $\boldsymbol{C}$.

Line 3 The i-th clustering algorithm C_i is taken from the set of clustering algorithms $\boldsymbol{C}$.

Line 4 The i-th ML model family F_i is created to be used as a result repository. For example, clustering models and supervised learning models are stored in the i-th ML model family.

Line 5 The function *Perform Clustering*(.) is executed. Note that Line 6 is explained after the explanation of the sub-functions in Algorithm 1.

Line 8 The function *Perform Clustering*(.) aims to set a clustering hyperparameter (i.e., the number of clusters) to each clustering algorithm.

Line 9 This function iterates the function *Perform Clustering(CL) Algorithm(Alg)*(.) in parallel. To do that, an index j is taken from 1 to the number of the set of hyperparameters $\boldsymbol{H}$ in the i-th clustering algorithm C_i. The index j denotes a hyperparameter used in the CL algorithm.

Line 10 The i-th CL algorithm is set with the hyperparameter H_j.

Line 11 The function *Perform Cl Alg*(.) is executed.

Line 14 The function *Perform Cl Alg*(.) aims to execute each clustering algorithm and prepare for the supervised learning algorithms.

Line 15 This function executes the clustering algorithm $C_{i,j}$ using the training data $\boldsymbol{D}_X$ corresponding to the clustering variables $\boldsymbol{V}$. Note that the training data which is not included in the clustering variables is ignored. The clustering model $CM_{i,j}$, then, is resulted from it.

Line 16 The clustering model $CM_{i,j}$ is assigned to the ML model family $F_{i,j}$.

Line 17 The clustered data CD_{XY} are taken from $\boldsymbol{D}_X$ and $\boldsymbol{D}_Y$ using the clustering model $CM_{i,j}$.

Line 18 This function iterates the function *Perform Supervised Learning*(.) in parallel. To do that, an index k is taken from 1 to the number of clustered data CD_{XY}. The index k denotes the k-th clustered data in the clustered data CD_{XY}.

Line 19 The k-th clustered data are taken from the clustered data CD_{XY}.

Line 20 The function *Supervised Learning*(.) is executed.

Line 23 The function *Supervised Learning*(.) aims to execute each supervised learning (SL) algorithm and return the evaluation score of a learned SL model.

Line 24 This function iterates in parallel from 1 to the set of supervised learning algorithms $\boldsymbol{S}$. The index l denotes the l-th SL algorithm.

Line 25 The l-th SL algorithm is taken from the set of supervised learning algorithms $\boldsymbol{S}$.

Line 26 The k-th clustered data are used to be split into the training data $TD_{XY,k}$ and the validation data $VD_{XY,k}$ using the K-Fold Cross-Validation (e.g., K = 5). The training data are used for machine learning, while the validation data are used for evaluation of a learned machine learning model.

Line 27 The l-th SL algorithm is executed using the training data $TD_{XY,k}$. The SL model SM_l is, then, generated.

Line 28 The SL model SM_l is used for prediction using the validation data $VD_{XY,k}$. The l-th prediction average score from the cross validation is stored. For this validation, various performance evaluation metrics

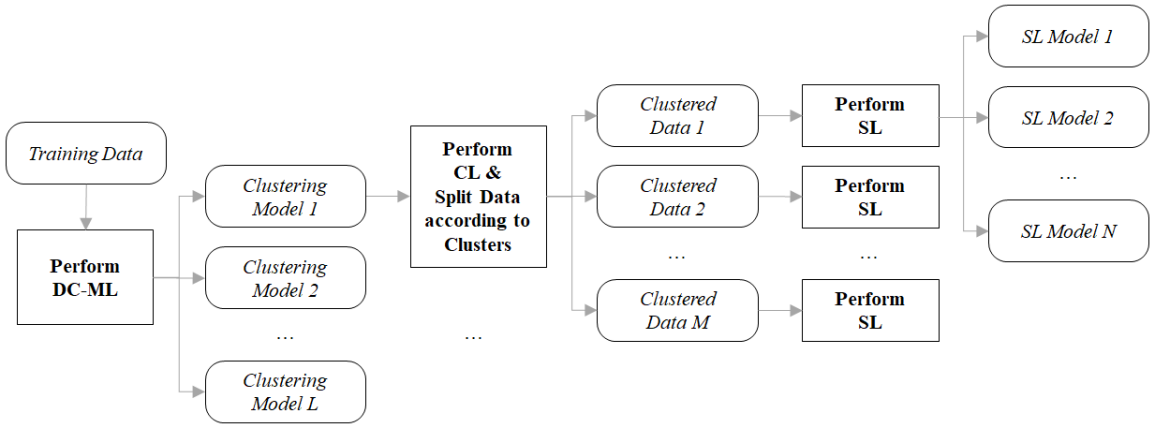


FIGURE 4. Concept of data clustering based machine learning (DC-ML).

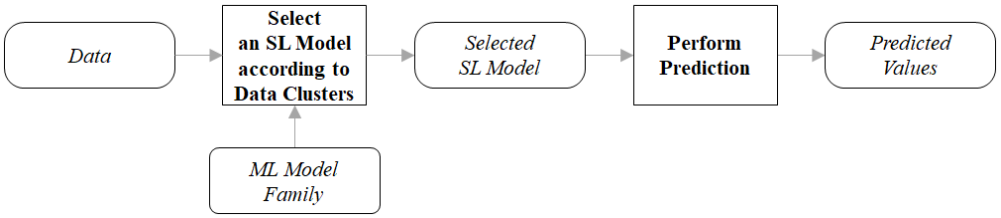


FIGURE 5. Prediction using an ML model family.

(e.g., R-Squared Score and Mean Square Error) can be used.

Line 29 The high-scored SL model F_{i,j,k^*} is selected using the set of the prediction average scores.

Line 30 The high-scored SL model F_{i,j,k^*} is stored in the set of the high-scored SL models F_{i,j,K^*}.

Line 22 The average score for the high-scored SL models in F_{i,j,K^*} is calculated. It is, then, assigned to $F_{i,j,avg}$.

Line 23 The average score $F_{i,j,avg}$ is stored in the set of the average scores $F_{i,J,avg}$.

Line 12 The high-scored clustering model which has the j^*-th hyperparameter is selected and it is assigned to F_{i,j^*}.

Line 13 The average score $F_{i,j^*,avg}$ of the high-scored clustering model F_{i,j^*} is stored in the set of the average scores $F_{I,j^*,avg}$.

Line 6 The high-scored i-th clustering model is selected using the set of the average scores $F_{I,j^*,avg}$.

Line 7 This algorithm outputs the high-scored ML model family F_{i^*,j^*,K^*} containing the high-scored i^*-th clustering model with the j^*-th hyperparameter and the set of high-scored SL models K^*.

We consider the time complexity of this algorithm in terms of the Big O. In this analysis, the time complexity of each machine learning algorithm is excluded, because it is beyond the scope of this research. In the algorithm, there exist four iterations (i.e., Lines 2, 9, 18, and 24), so the time complexity is $O(|\boldsymbol{C}|\times|C_i.H|\times|CD_{XY}|\times|\boldsymbol{S}|)$, where $\boldsymbol{C}$ denotes the set of clustering algorithms, $C_i.H$ denotes the set of hyperparameters of the i-th clustering algorithm, CD_{XY} denotes the clustered data, and $\boldsymbol{S}$ denotes the set of supervised learning algorithms. It seems like this is the computationally expensive operation. For example, for three clustering algorithms, three hyperparameters for each clustering algorithm, two clustered data, and three supervised learning algorithms, 54 processing tasks in total are required. However these iterations can be parallelizable, so in practice, actual operating time can be significantly reduced by using multithreading and/or multiprocessing. For example, if there are 54 multiprocessors, the total computing time can be the sum of the maximum processing times of a clustering algorithm and a supervised learning algorithm.

In addition, this paper presents a DC-ML software that was implemented in the Python programming language. The most recent version of the DC-ML software is available online at the DC-ML GitHub repository (https://github.com/pcyoung75/DC-ML).

IV. EXPERIMENTS IN THE SPM

In this section, we introduce two experiments to evaluate the predictive accuracy of the DC-ML algorithm. In this paper, the predictive accuracy means how correctly the models learned by the ML algorithms are mapped to a test data set. Specifically, a coefficient of determination (see Equation 14) is used for comparison between ML models. The experiments aim to find high-scored ML models for roll force and plate

TABLE 2. Selected variables in the real data.

Name	Description
Plate Thickness	The thickness of plate measured by a laser
Mill Modulus	The mill stand coefficient of vertical expansion due to repulsive force of material during rolling
Roll Force	The measure of repulsive force of material during rolling
Roll Gap	The set value of roll gap by screw down of the mill
Roll Gap Adjustment	The total adjustment value of roll gap for various inaccurate numbers of material size, material strength, material temperature, roll crown, etc
Type of Temperature Controlled Rolling	The type of rolling method to adjust temperature for proper metallurgical transformation
Material Strength at Rolling Temperature	This item is not measured in the factory.
Rolling Speed	The speed of work roll surface
Quantity of Material Deformation by Rolling	This item is not measured in the factory.
Material Strength	The mechanical yield strength for the composition of material
Material Temperature	The temperature of material at the time of rolling
Thickness Reduction	The quantity of thickness reduction during a pass of rolling
Width Reduction	The quantity of width reduction during a pass of rolling
Planned Plate Thickness	The planned target thickness of plate after rolling
Plate Width	The calculated width of plate after rolling
Roll Crown	The measure of the convex contour of roll

thickness predictions in each rolling pass using four existing ML algorithms (Gradient Boosting Regression (GB), Random Forest Regression (RF), Gaussian Process Regression (GP) and, CLG-BN (CG)) and the DC-ML algorithm.

For these two experiments, we performed four steps: (1) *acquiring real data*, (2) *developing a causal model*, (3) *performing machine learning*, and (4) *testing the prediction*. In the acquiring real data step, the real data for machine learning are collected from a target factory. In the causal model development step, a causal model, representing causal relationships between variables, is defined regarding the steel plate rolling factory. Such a causal model enables machine learning engineers to select a best structure (including features) of ML models. In the machine learning step, candidate ML models are trained using ML algorithms (including DC-ML) and the training data set from the target factory. In the test step, the learned ML models are evaluated using the test data set. Specifically, the roll force and plate thickness in each rolling pass (e.g., $PS_1, PS_2, \ldots, PS_N$) are predicted and then evaluated in terms of the accuracy.

The following subsection introduces each step of the experiment in detail. This experiment was performed on a 3.50GHz Intel Core i7-5930K processor with a 96 GB memory. Through these experiments, we determined two high-scored ML models that can be utilized in the operation of the SPM control systems.

A. ACQUIRING REAL DATA

The target factory contains several sensors and actuators to operate the rolling mill and other facilities (e.g., reheating furnaces and hot levelers). Factory data from these facilities are stored in real time on a main computer. For this research, some sample data, containing 4334 pass data cases, were used. Each pass data contained several sensor and actuator parameters (e.g., roll force, roll gap, and temperature) and their values. These parameters can be found in Table 2. For example, a plate production is scheduled with 18 rolling passes in which each rolling pass data are generated in the rolling mill operation (i.e., 18 pass data for one plate production). The last pass data contain the specification of the final results (e.g., the final production thickness of the plate). For each rolling pass, the values of these parameters were distributed in various ranges. For example, the input thickness of a plate before the rolling mill operation was around 272 millimeters, while the output thickness after the operation was around 17 millimeters.

B. DEVELOPING A CAUSAL MODEL FOR THE STEEL PLATE ROLLING FACTORY

Based on theoretical analysis of the thickness reduction process by rolling mills (see Subsections 2.A and 2.B), a causal model was developed by subject-matter experts in terms of a SPM control system, managing control values for a SPM. The causal model was used to identify main features and relationships between such features, so that machine learning engineers in this research comprehensively could understand the domain problem and situation, and could develop seamlessly machine learning models. Also, for machine learning, the causal model was used for feature engineering, in which features of data were selected. Usually, machine learning

Algorithm 1 Data Clustering based ML

Input: A training data set for predictor variables $\boldsymbol{D}_X$
Input: A training data set for a target variable $\boldsymbol{D}_Y$
Input: A set of clustering algorithms $\boldsymbol{C}$
Input: A set of supervised learning algorithms $\boldsymbol{S}$
Input: A set of clustering variables $\boldsymbol{V}$
Output: A high-scored ML model family F_{i^*,j^*,K^*}

1 **Function** *Run* ($\boldsymbol{D}_X$, $\boldsymbol{D}_Y$, $\boldsymbol{C}$, $\boldsymbol{S}$, $\boldsymbol{V}$)
2 **do in parallel for** $i \leftarrow 1$ **to** $|\boldsymbol{C}|$
3 $C_i \leftarrow$ have i-th clustering algorithm from $\boldsymbol{C}$
4 $F_i \leftarrow$ create an empty i-th ML model family
5 *Perform Clustering* ($\boldsymbol{D}_X$, $\boldsymbol{D}_Y$, C_i, F_i, $\boldsymbol{S}$, $\boldsymbol{V}$)
6 $F_{i^*} \leftarrow$ select the high-scored i-th clustering model using the average scores $F_{I,j^*,avg}$ (see Line 13)
7 **return** F_{i^*,j^*,K^*}
8 **Function** *Perform Clustering* ($\boldsymbol{D}_X$, $\boldsymbol{D}_Y$, C_i, F_i, $\boldsymbol{S}$, $\boldsymbol{V}$)
9 **do in parallel for** $j \leftarrow 1$ **to** $|C_i.H|$
10 $C_{i,j} \leftarrow$ set i-th clustering algorithm C_i with a candidate hyperparameter H_j
11 *Perform CL Alg* ($\boldsymbol{D}_X$, $\boldsymbol{D}_Y$, $C_{i,j}$, $F_{i,j}$, $\boldsymbol{S}$)
12 $F_{i,j^*} \leftarrow$ select the high-scored i-th clustering model with the j^* hyperparameter using the average scores $F_{i,J,avg}$ (see Line 23)
13 $F_{I,j^*,avg} \leftarrow F_{I,j^*,avg} \cup \{F_{i,j^*,avg}\}$
14 **Function** *Perform CL Alg* ($\boldsymbol{D}_X$, $\boldsymbol{D}_Y$, $C_{i,j}$, $F_{i,j}$, $\boldsymbol{S}$, $\boldsymbol{V}$)
15 $CM_{i,j} \leftarrow$ execute the clustering algorithm $C_{i,j}$ using the training data $\boldsymbol{D}_X$ associated with the variables $\boldsymbol{V}$ to get the clustering model $CM_{i,j}$
16 $F_{i,j} \leftarrow CM_{i,j}$
17 $CD_{XY} \leftarrow$ get the clustered data CD_{XY} from $\boldsymbol{D}_X$ and $\boldsymbol{D}_Y$ using the clustering model $CM_{i,j}$
18 **do in parallel for** $k \leftarrow 1$ **to** $|CD_{XY}|$
19 $CD_{XY,k} \leftarrow$ have k-th clustered data from CD_{XY}
20 *Perform Supervised Learning* ($CD_{XY,k}$, $F_{i,j,k}$, $\boldsymbol{S}$)
21 $F_{i,j,avg} \leftarrow$ calculate the average score for the SL models in F_{i,j,K^*} and store it into $F_{i,j,avg}$
22 $F_{i,J,avg} \leftarrow F_{i,J,avg} \cup \{F_{i,j,avg}\}$
23 **Function** *Perform Supervised Learning* ($CD_{XY,k}$, $F_{i,j,k}$, $\boldsymbol{S}$)
24 **do in parallel for** $l \leftarrow 1$ **to** $|\boldsymbol{S}|$
25 $S_l \leftarrow$ have l-th SL algorithm from $\boldsymbol{S}$
26 $TD_{XY,k}$, $VD_{XY,k} \leftarrow$ perform the K-Fold split to get the training data $TD_{XY,k}$ and the validation data $VD_{XY,k}$ from $CD_{XY,k}$;
27 $SM_l \leftarrow$ perform the SL algorithm S_l using $TD_{XY,k}$ to get the SL model SM_l
28 $\boldsymbol{avgScore}_l \leftarrow$ perform the prediction using SM_l and $VD_{XY,k}$ to get an l-th average score
29 $F_{i,j,k^*} \leftarrow$ find the high-scored SL model using the set of scores $\boldsymbol{avgScore}$ and put it into F_{i,j,k^*}
30 $F_{i,j,K^*} \leftarrow F_{i,j,K^*} \cup \{F_{i,j,k^*}\}$

engineers lack the domain knowledge to which they are assigned. The subject-matter experts also do not have much knowledge about machine learning algorithms. The causal model could help both experts to understand the target situation and develop the ML models.

There were a number of factors (i.e., predictor variables) that might have affected the plate thickness and the roll force (i.e., target variables). However, some predictor variables can be negligible, because of redundancy and small influence to the target variables. For this, we selected candidate control and noise factors of the SPM control system, and determined the relationships between these factors using the theory of the rolling process in Subsection 2.B.

Figure 6 shows the causal model in this paper. This causal model was developed in terms of *Plate Thickness* and its causal factors (e.g., *Mill Modulus* and *Temperature*). For the SPM control system, the first-order control factors are *Roll Gap* and *Roll Gap Adjustment*, while the first-order noise factors are *Mill Modulus* and *Roll Force*. The causal model shows also the second and third-order noise factors for the SPM control system. In addition, there are two factors, represented by the dashed boxes (i.e., *Material Strength at rolling temperature* and *Quantity of material deformation by rolling*), for which corresponding data do not exist. These two factors are included in the causal model, because by doing this, hidden factors can be displayed more explicitly.

Table 2 shows all the features (or variables) used in this paper. The total 16 features were identified through this step. For example, *Plate Thickness* in Table 2 is the thickness of a plate measured by a laser. *Planned Plate Thickness* is the planned target thickness of a plate after each rolling.

C. PERFORMING MACHINE LEARNING

Initially, we considered various machine learning algorithms (e.g., decision tree, support vector machine, and deep learning), however since they did not result in any noticeably better performance compared to the results from the four algorithms in Subsection 2.C, we did not include them in this experiment. For the roll force and plate thickness predictions, the four algorithms in Subsection 2.C and the DC-ML algorithm in Subsection 3 were used to learn each ML model of the corresponding ML algorithm.

To perform these ML algorithms, identifying predictor variables and target variables was required. From the causal model in Figure 6, the predictor variables and the target variables were identified by the subject-matter experts. Table 3 shows the variables for the roll force prediction, while Table 4 shows the variables for the plate thickness prediction.

For DC-ML, three clustering algorithms (*Gaussian Mixture* [53], *Birch* [54], and *Mini Batch K-Means* [33]) were used as input. For each clustering algorithm, 2∼7 cluster numbers were set as the candidate hyperparameters. Note that eight or more cluster numbers can be set, but an experiment

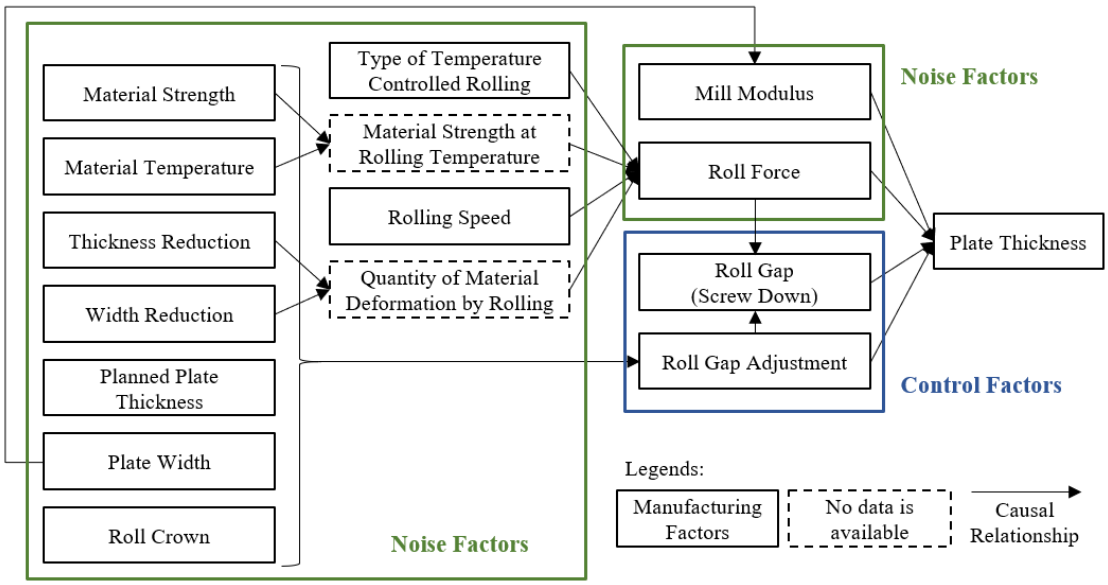


FIGURE 6. Causal model for steel plate rolling factory.

TABLE 3. Selected variables for roll force prediction.

Predictor Variables	Target Variable
Mill Modulus Material Temperature Thickness Reduction Width Reduction Material Strength Roll Gap Planned Plate Thickness	Roll Force

TABLE 4. Selected variables for plate thickness prediction.

Predictor Variables	Target Variable
Mill Modulus Material Temperature Thickness Reduction Width Reduction Material Strength Rolling Speed Roll Force Roll Gap Roll Gap Adjustment	Plate Thickness

takes a lot of time. Furthermore, as we will see in Section 5, a four-clusters model yields the best result.

The data in Subsection 4.A were randomly divided in 90% of the training data and 10% of the test data. Each ML algorithm test was repeated up to 20 times. When performing DC-ML, the training and validation data were randomly selected using the 5-fold cross-validation and Mean Absolute Error was used for the validation of candidate models.

The following steps summarizes the experiment process for the roll force and plate thickness prediction in detail.

Step 1. The training data of 90% and test data of 10% were randomly taken from the real data set (4334 cases) according to the experiment type (the roll force prediction or the plate thickness prediction)

Step 2. The DC-ML algorithm was used to learn an ML model using four inputs: (1) the training data, (2) the set of clustering algorithms (Gaussian Mixture, Birch, and Mini Batch K Means), (3) The set of supervised learning algorithms (Random Forest Regression (RF), Gradient Boosting Regression (GB), Gaussian Process Regression (GP), and CLG-BN (CG)), and (4) the one clustering variable (Material Strength). Each input clustering algorithm was set with 2 to 7 cluster numbers as hyperparameters. In this setting, the DC-ML algorithm was performed 20 times for each cluster numbers.

Step 3. The training data of 90% were reused to learn each of four ML models (RF, GB, GP, and CG).

Step 4. After machine learning, the DC-ML model in Step 2 and the four learned models in Step 3 were evaluated using the test data of 10%.

D. TESTING PREDICTION

To evaluate the five ML models from the previous subsection, the coefficient of determination, called R^2 score (Equation 14), were used. Note that 1 of R^2 score means that the model perfectly predicted the results without an error. And a negative R^2 score can occur, when poorly predicting the results.

$$R^2 = 1 - \frac{SSE}{SST}, \tag{14}$$

TABLE 5. Overall average R^2 score in roll force prediction.

ML	Average	Standard Deviation
Gradient Boosting Regression	0.7381	0.0297
Random Forest Regression	0.8475	0.0193
Gaussian Process Regression	0.2066	0.0333
Conditional Linear Gaussian BN	0.8632	0.0143
DC-ML	**0.8828**	**0.0117**

where SSE denotes the sum of squared errors

$$SSE = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 \tag{15}$$

and SST denotes the total sum of squares

$$SST = \sum_{i=1}^{n} (y^{(i)} - \mu_y)^2 \tag{16}$$

in which n denotes the total number of the data cases and y denotes the actual target value in a case, $\hat{y}$ denotes prediction, and μ_y denotes the average of actual target values.

V. RESULTS AND DISCUSSION

In this section, we evaluate the five machine learning algorithms and present the lessons learned regarding the application of machine learning in SPM.

A. EVALUATION FOR MACHINE LEARNING ALGORITHMS

For the two experiments (the roll force and plate thickness predictions in SPM), two high-scored ML model families were selected. These two models contained the same four clusters. Such ML model families can be called a *four-cluster ML model family*. In the following two subsections, the DC-ML models mean the four-cluster ML model family.

1) EVALUATION FOR ROLL FORCE PREDICTION

In the roll force prediction, the four-cluster ML model family showed better results than the four regression models (GB, RF, GP, and CG). Table 5 shows an *overall average* R^2 score of each of the five ML algorithms. The R^2 score denotes the prediction accuracy (Equation 14), evaluated by comparing the actual values and the predicted values. The overall average R^2 score means the average of the R^2 scores from 20 tests

In the prediction results, the ML algorithms Gradient Boosting Regression, Random Forest Regression, and CLG-BN resulted in relatively lower scores than the ML algorithm DC-ML. DC-ML predicted the roll force with the highest accuracy (0.8828) and precision (0.0117). Among the four algorithms except DC-ML, Conditional Linear Gaussian showed the highest result (0.8632), while Gaussian Process Regression showed the lowest result (0.2066).

Figure 7 shows a box-plot chart corresponding to data in Table 5. In the figure, the ML algorithm Gaussian Process Regression was excluded to investigate precisely the results from the other algorithms.

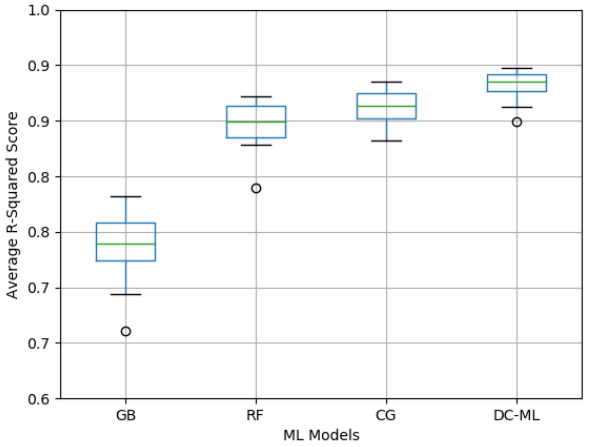


FIGURE 7. Overall average R^2 score for roll force prediction.

TABLE 6. Percentage of selected clustering algorithms in the 20 tests for roll force prediction.

Gaussian Mixture	Birch	Mini Batch K-Means
25%	10%	65%

TABLE 7. Percentage of selected supervised learning algorithms in the ML model families for roll force prediction.

Gradient Boosting Regression	Random Forest Regression	Gaussian Process Regression	Conditional Linear Gaussian BN
0%	25%	0%	75%

In the 20 times test, the 20 high-scored ML model families were learned using DC-ML. For each test, a high-scored ML model family contained a different clustering model. The input set of clustering algorithms were Gaussian Mixture, Birch, and Mini Batch K-Means. Table 6 shows the percentage of selected clustering algorithms in the 20 tests. Gaussian Mixture, Birch, and Mini Batch K-Means were selected with 25 percent, 10 percent, and 65 percent, respectively.

For each cluster of the four clusters in the 20 high-scored ML model families, one of the four supervised learning models was selected. Table 7 shows the percentage of selected supervised learning algorithms in the ML model families. For example, Random Forest Regression was selected with 25 percent, while CLG-BN was selected with 75 percent. Among the four supervised learning algorithms, CLG-BN was shown as a best algorithm. And this result is consistent with the results in Table 5.

2) EVALUATION FOR PLATE THICKNESS PREDICTION

Like the previous subsection, high-scored ML model families for the plate thickness prediction were learned using DC-ML. Table 8 shows the overall average R^2 score of each of the five ML algorithms.

For the five ML algorithms, the prediction results look similar around 0.999 of the overall average R^2 score. However, the SPM control systems require a high level of accu-

TABLE 8. Overall average R^2 scores in plate thickness prediction.

ML	Average	Standard Deviation
Gradient Boosting Regression	0.9996499	0.0000551
Random Forest Regression	0.9999032	0.0001174
Gaussian Process Regression	0.9999957	0.0000009
Conditional Linear Gaussian BN	0.9999957	0.0000009
DC-ML	**0.9999959**	**0.0000008**

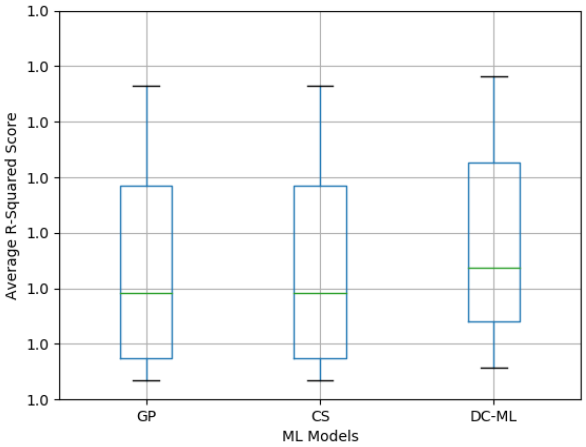


FIGURE 8. Overall average R^2 score for plate thickness prediction.

TABLE 9. Percentage of selected clustering algorithms in the ML model families for plate thickness prediction.

Gaussian Mixture	Birch	Mini Batch K-Means
20%	55%	25%

racy, because it directly influences the quality of the final product (i.e., a steel plate). The higher prediction accuracy is significant in this domain. The ML algorithms Gradient Boosting Regression and Random Forest Regression resulted in relatively lower scores than the ML algorithms Gaussian Process Regression, Conditional Linear Gaussian BN, and DC-ML. DC-ML predicted the plate thickness with slightly higher accuracy (0.9999959) and precision (0.0000008).

Figure 8 shows a box-plot chart corresponding to data in Table 8. In Figure 8, the ML algorithms Gradient Boosting Regression and Random Forest Regression were excluded to investigate precisely the results from Gaussian Process Regression (GP), CLG-BN (CG), and DC-ML.

Table 9 shows the percentage of selected clustering algorithms in the high-scored ML model families. Of the clustering algorithms Gaussian Mixture, Birch, and Mini Batch K-Means, Gaussian Mixture was selected with 20 percent, Birch was selected with 55 percent, and Mini Batch K-Means was selected with 25 percent.

In addition, Table 10 shows the percentage of selected supervised learning algorithms in the ML model families. This result is consistent with the results in Table 8. For example, Gaussian Process Regression and Conditional Linear Gaussian BN were selected with 67 and 33 percent, respectively, while Gradient Boosting Regression and Random Forest Regression were not selected as shown in their low scores in Table 8.

TABLE 10. Percentage of selected supervised learning algorithms in the ML model families for plate thickness prediction.

Gradient Boosting Regression	Random Forest Regression	Gaussian Process Regression	Conditional Linear Gaussian BN
0%	0%	67%	33%

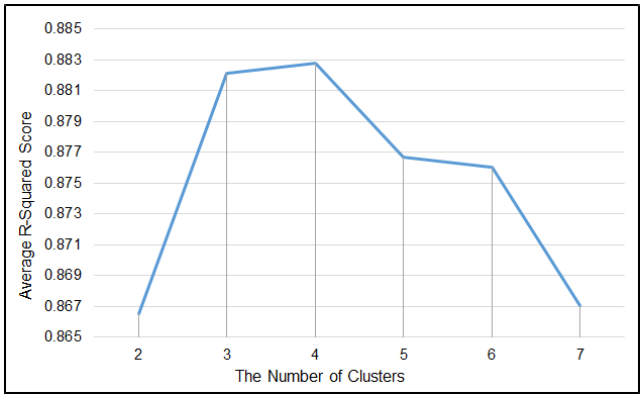


FIGURE 9. Overall average R^2 scores for roll force prediction over clusters.

B. LESSONS LEARNED

This subsection introduces the lessons learned from this research to help researchers related to a smart factory make better decision, when applying machine learning.

- **Data Clustering for Smart Manufacturing**

  Smart manufacturing aims small-quantity batch production for various products. The wide range of products generates a variety of data. In such a case, using a single ML model may not be able to achieve effective results. Instead, the approach of using multiple ML models can provide better performance, because the data in this case contains separable sub-data. For example, in this paper, the four-cluster ML model family (i.e., the multiple ML model approach) showed better results than the approach of using the single ML model.

- **Cluster Numbers and Data Size**

  The performance of DC-ML is mainly influenced by the quality of clusters. In data of a fixed size, as the number of clusters increases, the number of available data for supervised learning decreases. The number of data influences the quality of the supervised learning model. Therefore, it is required to find the appropriate number of clusters. Figure 9 depicts the overall average R^2 scores for the roll force prediction over 2∼7 cluster numbers in the experiment of Subsection 5.A.

  As the number of clusters increases from 2 to 7, the score increases and decreases after the four clusters.

The figure represents a typical correlation between the cluster numbers (or data size) and the model quality. To improve the performance of DC-ML, a method of recommending the appropriate number of clusters is required. To address this issue, a simple grid search can be used. However, as the number of clustering variables increases, the total number of the searching space can exponentially increase. We leave this for future research.

- **Usefulness of Causal Models**

  Although it is not trivial to derive a causal model (e.g., Figure 6) from the target domain, it can help one understand aspects of that field, find weak and/or strong influencing factors, and utilize existing domain knowledge (e.g., physical and chemical characteristics) to construct ML models. Understanding the target domain using the causal model enables us to determine suitable candidates of machine learning models and algorithms in advance so that we can efficiently deliver the domain knowledge to ML engineers.

- **Static and Dynamic ML Models**

  If data are sequential in nature, a dynamic ML model (e.g., Recurrent Neural Network (RNN) [55], [56] and Long Short-Term Memory (LSTM) [57]) is usually required. However, by changing dynamic data to static data, a static ML model, representing just one snapshot in time, can be applied. In this research, we found that the current manufacturing factors are influenced only by factors in the previous pass (i.e., a first-order Markov assumption, a factor at a time n only depends on a factor at a time $n-1$). In this case, simply combining the current pass data with the previous pass data is sufficient to train the static ML model.

- **Missing Data and Data Precision**

  In our experience of applying machine learning to smart factories, oftentimes we have encountered a missing-data situation in which proper data are missed or the precision of the acquired data are too low to apply machine learning. For machine learning, collecting the right data is the most imperative task that should be performed in the data acquisition phase. It is highly recommended to collect high-precision data. However, it will be costly. Therefore, finding right prediction level according to analysis goals is a critical task.

VI. CONCLUSION

In this paper, we presented ML technologies in a steel plate production line. We focused on finding high-scored ML algorithms which can be used for the roll force and plate thickness prediction at each rolling pass, so that one can find the best control conditions to produce high-quality steel plate products. In addition, the ML approach in this paper can reduce a sensor cost as well as its operational cost. In our experiment, DC-ML shows the acceptable results for the roll force and plate thickness prediction.

The idea behind this paper can also be used to apply other operations in a smart factory. In the era of Big Data, unused data in manufacturing lines are overflowing and sleeping. The prediction capability of machine learning with such data can be utilized for replacing existing facilities, devices, and sensors in the manufacturing lines. By doing so, the operational cost can be significantly reduced. Especially, DC-ML has characteristics suitable for smart manufacturing, aiming small-quantity batch production for various products, because it can provide multiple ML models according to different kinds of products in a same category. In this paper, we only focused on the operation of steel plate rolling smart factory. Future work will consider to apply the approach in this paper to other facilities and other smart factories.

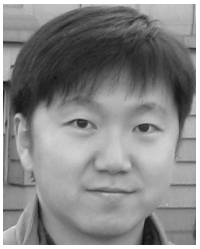
ACKNOWLEDGMENT

The authors would like to thank Dr. Sung Tae Kim for his statistical analysis in this research that provided insight into the understanding of the target system. They would also like to thank Dr. Shou Matsumoto, Mr. Hang Seok Choi, and Mr. Dong Jin Lee for their insightful comments on this research, and Mr. JuByung Ha for his contributions in data engineering.

REFERENCES

[1] S. M. L. Coalition, "Implementing 21st century smart manufacturing," in *Proc. Workshop Summary Rep.*, 2011, pp. 1–42.
[2] J. Lee, H.-A. Kao, and S. Yang, "Service innovation and smart analytics for industry 4.0 and big data environment," *Procedia CIRP*, vol. 16, pp. 3–8, Jan. 2014.
[3] Y. Lu, K. C. Morris, and S. Frechette, "Current standards landscape for smart manufacturing systems," *Nat. Inst. Standards Technol.*, vol. 8107, p. 39, Feb. 2016.
[4] C. Y. Park, K. B. Laskey, S. Salim, and J. Y. Lee, "Predictive situation awareness model for smart manufacturing," in *Proc. 20th Int. Conf. Inf. Fusion (Fusion)*, Jul. 2017, pp. 1–8.
[5] J. Li, H. Deng, and W. Jiang, "Secure vibration control of flexible arms based on operators' behaviors," in *Proc. Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage*. Cham, Switzerland: Springer, 2017, pp. 420–431.
[6] H. Lee and J. Lee, "Development concepts of smart service system-based smart factory (4SF)," in *Proc. INCOSE Int. Symp.*, Jul. 2018, vol. 28, no. 1, pp. 1153–1169.
[7] G. Qiao and B. A. Weiss, "Quick health assessment for industrial robot health degradation and the supporting advanced sensing development," *J. Manuf. Syst.*, vol. 48, pp. 51–59, Jul. 2018.
[8] K. S. Kiangala and Z. Wang, "Initiating predictive maintenance for a conveyor motor in a bottling plant using industry 4.0 concepts," *Int. J. Adv. Manuf. Technol.*, vol. 97, nos. 9–12, pp. 3251–3271, Aug. 2018.
[9] Y. J. Qu, X. G. Ming, Z. W. Liu, X. Y. Zhang, and Z. T. Hou, "Smart manufacturing systems: State of the art and future trends," *Int. J. Adv. Manuf. Technol.*, vol. 103, nos. 9–12, pp. 3751–3768, Aug. 2019.
[10] A. D. Landmark, E. Arica, B. Kløve, P. F. Kamsvåg, E. A. Seim, and M. Oliveira, "Situation awareness for effective production control," in *Proc. IFIP Int. Conf. Adv. Prod. Manage. Syst.* Cham, Switzerland: Springer, 2019, pp. 690–698.
[11] T. Nkonyana, Y. Sun, B. Twala, and E. Dogo, "Performance evaluation of data mining techniques in steel manufacturing industry," *Procedia Manuf.*, vol. 35, pp. 623–628, Jan. 2019.
[12] S. Guo, J. Yu, X. Liu, C. Wang, and Q. Jiang, "A predicting model for properties of steel using the industrial big data based on machine learning," *Comput. Mater. Sci.*, vol. 160, pp. 95–104, Apr. 2019.

[13] K. J. Åström, T. Hägglund, C. C. Hang, and W. K. Ho, "Automatic tuning and adaptation for pid controllers—A survey," *Control Eng. Pract.*, vol. 1, no. 4, pp. 699–714, 1993.
[14] S. Bennett, "The past of PID controllers," *Annu. Rev. Control*, vol. 25, pp. 43–53, Jan. 2001.
[15] K. J. Åström and T. Hägglund, *Advanced PID Control*. Research Triangle Park, NC, USA: ISA-The Instrumentation, Systems, and Automation Society, 2006.
[16] X. Zhang, X. Yao, Q. Wu, and D. Li, "The application of generalized predictive control to the HAGC," in *Proc. 5th Int. Conf. Fuzzy Syst. Knowl. Discovery*, vol. 1, Oct. 2008, pp. 444–447.
[17] A. S. Karandaev, A. A. Radionov, V. R. Khramshin, I. Y. Andryushin, and A. G. Shubin, "Automatic gauge control system with combined control of the screw-down arrangement position," in *Proc. 12th Int. Conf. Actual Problems Electron. Instrum. Eng. (APEIE)*, Oct. 2014, pp. 88–94.
[18] Z. Zhang and W. Ding, "A new anti-disturbance strategy of automatic gauge control for small workroll cold reversing mill," in *Proc. IEEE Adv. Inf. Manage., Commun., Electron. Automat. Control Conf. (IMCEC)*, Oct. 2016, pp. 2004–2008.
[19] S. Wang, "Real-time neurofuzzy control for rolling mills," Ph.D. dissertation, Dept. Mech. Eng., Univ. Wollongong, Wollongong, NSW, Australia, 1999.
[20] C. V. Negoita, "Fuzzy sets," *Fuzzy Sets Syst.*, vol. 133, no. 2, p. 275, Jan. 2003.
[21] X. Wang, Y. Xiao, and D. Zhang, "The design of mill automatic gauge control system based on the fuzzy proportion integral differential controller," in *Proc. 5th Int. Conf. Fuzzy Syst. Knowl. Discovery*, vol. 3, Oct. 2008, pp. 249–253.
[22] V. Ginzburg, *Steel-Rolling Technology: Theory and Practice*. New York, NY, USA: Marcel, 1989.
[23] D. M. Lee and S. G. Choi, "Application of on-line adaptable neural network for the rolling force set-up of a plate mill," *Eng. Appl. Artif. Intell.*, vol. 17, no. 5, pp. 557–565, Aug. 2004.
[24] F. Zhang, Y. Zhao, and J. Shao, "Rolling force prediction in heavy plate rolling based on uniform differential neural network," *J. Control Sci. Eng.*, vol. 2016, pp. 1–9, Jun. 2016.
[25] S. Rath, A. P. Singh, U. Bhaskar, B. Krishna, B. K. Santra, D. Rai, and N. Neogi, "Artificial neural network modeling for prediction of roll force during plate rolling process," *Mater. Manuf. Process.*, vol. 25, nos. 1–3, pp. 149–153, Mar. 2010.
[26] M. Bagheripoor and H. Bisadi, "Application of artificial neural networks for the prediction of roll force and roll torque in hot strip rolling process," *Appl. Math. Model.*, vol. 37, no. 7, pp. 4593–4607, Apr. 2013.
[27] Z.-H. Wang, D.-Y. Gong, X. Li, G.-T. Li, and D.-H. Zhang, "Prediction of bending force in the hot strip rolling process using artificial neural network and genetic algorithm (ANN-GA)," *Int. J. Adv. Manuf. Technol.*, vol. 93, nos. 9–12, pp. 3325–3338, Dec. 2017.
[28] J. Liu, X. Liu, and B. T. Le, "Rolling force prediction of hot rolling based on GA-MELM," *Complexity*, vol. 2019, pp. 1–11, Jun. 2019.
[29] K. Esendağ, A. H. Orta, İ. Kayabaşı, and S. Ilker, "Prediction of reversible cold rolling process parameters with artificial neural network and regression models for industrial applications: A case study," *Procedia CIRP*, vol. 79, pp. 644–648, Jan. 2019.
[30] H. Wang, B. van Stein, M. Emmerich, and T. Bäck, "Time complexity reduction in efficient global optimization using cluster kriging," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, 2017, pp. 889–896.
[31] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, vol. 2, no. 3. Cambridge, MA, USA: MIT Press, 2006.
[32] F. Qiang, H. Shang-Xu, and Z. Sheng-Ying, "Clustering-based selective neural network ensemble," *J. Zhejiang Univ.-Sci. A*, vol. 6, no. 5, pp. 387–392, May 2005.
[33] J. A. Hartigan, *Clustering Algorithms*. Hoboken, NJ, USA: Wiley, 1975.
[34] W. L. Roberts, *Flat Processing of Steel*. New York, NY, USA: Marcel Dekker, 1988.
[35] V. B. Ginzburg and R. Ballas, *Flat Rolling Fundamentals*. Boca Raton, FL, USA: CRC Press, 2000.
[36] H. Yim, B. Joo, G. Lee, J. Seo, and Y. Moon, "A study on the roll gap set-up to compensate thickness variation at top-end in plate rolling," *Trans. Mater. Process.*, vol. 18, no. 4, pp. 290–295, 2009.
[37] Y.-H. Moon and J.-J. Yi, "Improvement of roll-gap set-up accuracy using a modified mill stiffness from gaugemeter diagrams," *J. Mater. Process. Technol.*, vol. 70, nos. 1–3, pp. 194–197, Oct. 1997.
[38] Y. M. Hwang and H. H. Hsu, "An investigation into the plastic deformation behavior at the roll gap during plate rolling," *J. Mater. Process. Technol.*, vol. 88, nos. 1–3, pp. 97–104, Apr. 1999.
[39] M. Deisenroth and J. W. Ng, "Distributed Gaussian processes," in *Proc. 32nd Int. Conf. Mach. Learn.*, in Proceedings of Machine Learning Research, vol. 37, F. Bach and D. Blei, Eds., Lille, France, Jul. 2015, pp. 1481–1490. [Online]. Available: http://proceedings.mlr.press/v37/deisenroth15.html
[40] T. K. Ho, "Random decision forests," in *Proc. 3rd Int. Conf. Document Anal. Recognit.*, vol. 1, Aug. 1995, pp. 278–282.
[41] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Pacific Grove, CA, USA: Brooks/Cole, 1984.
[42] L. Breiman, "Arcing classifiers," *Ann. Statist.*, vol. 26, pp. 40–123, Feb. 1996.
[43] R. E. Schapire, "The strength of weak learnability," *Mach. Learn.*, vol. 5, no. 2, pp. 197–227, 1990.
[44] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. ICML*, vol. 96, 1996, pp. 148–156.
[45] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*. Berlin, Germany: Springer, 2003, pp. 63–71.
[46] S. L. Lauritzen and N. Wermuth, "Graphical models for associations between variables, some of which are qualitative and some quantitative," *Ann. Statist.*, vol. 17, no. 1, pp. 31–57, Mar. 1989.
[47] W. Sun, C. Y. Park, and R. Carvalho, "A new research tool for hybrid Bayesian networks using script language," *Proc. SPIE*, vol. 8050, May 2011, Art. no. 80501Q.
[48] C. Y. Park, K. B. Laskey, P. C. G. Costa, and S. Matsumoto, "Message passing for hybrid Bayesian networks using Gaussian mixture reduction," in *Proc. 10th Int. Conf. Digit. Inf. Manage. (ICDIM)*, Oct. 2015, pp. 210–216.
[49] C. Y. Park, K. B. Laskey, P. C. G. Costa, and S. Matsumoto, "Gaussian mixture reduction for time-constrained approximate inference in hybrid Bayesian networks," *Appl. Sci.*, vol. 9, no. 10, p. 2055, 2019.
[50] A. Rencher, *Methods of Multivariate Analysis*. Hoboken, NJ, USA: Wiley, 2002.
[51] S. Matsumoto, R. N. Carvalho, M. Ladeira, P. C. G. da Costa, L. L. Santos, D. Silva, M. Onishi, E. Machado, and K. Cai, "UnBBayes: A Java framework for probabilistic models in AI," in *Java in Academia and Research*, K. Cai, Ed. Annerley, QLD, Australia: iConcept Press, 2011, ch. 9, p. 34.
[52] C. Y. Park, K. B. Laskey, P. C. G. Costa, and S. Matsumoto, "Multi-entity Bayesian networks learning for hybrid variables in situation awareness," in *Proc. 16th Int. Conf. Inf. Fusion*, 2013, pp. 1894–1901.
[53] G. Celeux and G. Govaert, "A classification EM algorithm for clustering and two stochastic versions," *Comput. Statist. Data Anal.*, vol. 14, no. 3, pp. 315–332, Oct. 1992.
[54] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *ACM SIGMOD Rec.*, vol. 25, no. 2, pp. 103–114, 1996.
[55] B. A. Pearlmutter, "Learning state space trajectories in recurrent neural networks," *Neural Comput.*, vol. 1, no. 2, pp. 263–269, Jun. 1989.
[56] C. L. Giles, G. M. Kuhn, and R. J. Williams, "Dynamic recurrent neural networks: Theory and applications," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 153–156, Mar. 1994.
[57] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.



CHEOL YOUNG PARK (Member, IEEE) has worked as a Research Associate with the C4I and Cyber Center, GMU. He is currently working as a Machine Learning Research Engineer with BAIES, LLC. At BAIES, he conducted the project about a collective intelligence multimodel integration platform, called Bayes Cloud. He has researched and developed machine learning algorithms for multientity Bayesian networks to support predictive situation awareness systems, such as a MSAW system for smart manufacturing, a PROGNOS system for maritime situation awareness, and a HERALD system for critical infrastructure defense. His researches were supported by funds from the Office of Naval Research, KEIT, and POSCO. He volunteers to teach artificial intelligence and software programming to high school students in Northern Virginia, USA.

JIN WOOG KIM currently runs a DEEP-IN Company, South Korea, and runs an auto-trading system for trading cryptocurrencies and FX based on AI engine using Bayesian deep learning. He is exploring, in reinforcement learning, how to discard existing historical data and learn new data effectively with only a small amount of computation, while data is constantly being input in real time. In particular, he has been researching neural network ensembles that use Bayesian reasoning to improve learning performance in transfer learning and share weights between models that have learned Dropout. His Ph.D. research focuses on the initial weighting of neural networks using prior probabilities and the relative performance improvement of deep learning models.

BOSUNG KIM received the bachelor's degree in material engineering from PNU. He has pursued a research agenda focusing on the Steel Plate Rolling Engineering Technology in steel making industry. He is currently working as the Junior Manager with POSCO Corporation. He has tried to improve accuracy and precision in the rolling control systems. He has researched and developed various micro control application systems in rolling mill for seven years.

JOONGYOON LEE has worked as a Researcher with DAEWOO Motor Corporation and a Chief Architect and the CEO with SE Technology Corporation. He has been working as a Professor of systems engineering with POSTECH University, since 2012. He has researched focusing on the application of the systems engineering technology for various industrial areas. He has researched and developed architectures of smart manufacturing systems, railway systems, plant systems, and various military systems. He has been serving INCOSE as a representative of the Korean Chapter. His Ph.D. research subject was a study on the process and tool for system requirements definition. He is also a member of the ISO/IEC JTC1 SC7 for software and systems engineering.