A Convolutional Neural Network for Fault Classification and Diagnosis in Semiconductor Manufacturing Processes

Ki Bum Lee, Sejune Cheon, and Chang Ouk Kim

***Abstract*—Many studies on the prediction of manufacturing results using sensor signals have been conducted in the field of fault detection and classification (FDC) for semiconductor manufacturing processes. However, fault diagnosis used to find clues as to root causes remains a challenging area. In particular, process monitoring using neural networks has been employed to only a limited extent because it is a black box model, making the relationships between input data and output results difficult to interpret in actual manufacturing settings, despite its high classification performance. In this paper, we propose a convolutional neural network (CNN) model, named FDC-CNN, in which a receptive field tailored to multivariate sensor signals slides along the time axis, to extract fault features. This approach enables the association of the output of the first convolutional layer with the structural meaning of the raw data, making it possible to locate the variable and time information that represents process faults. In an experiment on a chemical vapor deposition process, the proposed method outperformed other deep learning models.**

***Index Terms*—Fault detection and classification, fault diagnosis, convolutional neural network, deep learning, multivariate time-series data, semiconductor manufacturing.**

I. Introduction

IN SEMICONDUCTOR manufacturing, fault detection and classification (FDC) involves developing models for monitoring wafer manufacturing results with statistical and machine learning techniques using multivariate sensor signals (*i.e.*, data streams) collected during each wafer processing. High-performance FDC models detect wafer faults at an early manufacturing step and prevent the defectives from being released to the downstream steps. Thus, effective FDC implementation contributes to yield enhancement and cost reduction. Fault diagnosis, which is an analysis conducted subsequent to fault detection, determines the correct fault position on the data, thus providing key information for analyzing the root cause of a fault [1]. Considering that the detection of the root cause is typically conducted by offline analysis that relies on the empirical knowledge of experts, automatic fault diagnosis is an ultimate goal of data-driven fault analysis.

Relevant studies to date focusing on fault diagnosis in the field of semiconductor manufacturing are as follows. Nawaz *et al.* [2] proposed a Bayesian-network-based inference system for the etching process, therein considering the cause–effect relationships between root causes, equipment, and process parameters; the limitation of this method is that it is a knowledge-based approach requiring expert knowledge of the related field. Among data-driven approaches, Chien *et al.* [3] selected variables crucial to Hotelling's T^2 and squared prediction error charts, therein using contribution plots of multiway principal component analysis in a chemical process, and analyzed the association between faulty wafers and sensor variables using decision trees. There have also been extensive studies of FDC models using support vector machines (SVMs). Among them, Mahadevan and Shah [4] detected abnormal process behaviors in the semiconductor etching process using a one-class SVM. They also used SVM recursive feature elimination to determine and analyze the root cause of a fault occurrence.

This study proposes an advanced FDC model with automatic feature extraction and fault diagnosis functions, which are enabled by a convolutional neural network (CNN) proposed to suit the structure of the multivariate sensor signals from a semiconductor manufacturing process. The proposed FDC model is appropriate for chemical processes in wafer fabrication such as etching and chemical vapor deposition. In a process tool, recipe parameters, such as temperature, pressure, voltage currents, and chemical gas, work together in a chamber according to the condition specified in a process recipe. The condition may be violated due to unexpected chemical and mechanical disturbance factors. Then, the tool will not operate as desired and, as the result, the wafer in the chamber will not be manufactured normally. Advanced tools trace the states of recipe parameters in seconds using sensors. The proposed model finds sensor signals that cause wafer faults.

CNN is a deep learning model [5] evolved from artificial neural networks (ANNs), a type of machine learning algorithm that mimics the working mechanism of the human brain.

Manuscript received September 21, 2016; revised December 5, 2016; accepted February 25, 2017. Date of publication March 1, 2017; date of current version May 3, 2017. This work was supported in part by the Global Ph.D. Fellowship Program through the National Research Foundation of Korea (NRF) under Grant NRF-2015H1A2A1031081, in part by the Technology Innovation Program (Development of Big Data-Based Analysis and Control Platform for Semiconductor Manufacturing Plants) through the Ministry of Trade, Industry and Energy, South Korea, under Grant 10045913, and in part by the NRF through the Ministry of Science, ICT and Future Planning, South Korea, under Grant NRF-2016R1A2B4008337.

The authors are with the Department of Industrial Engineering, Yonsei University, Seoul 120-749, South Korea (e-mail: kimco@yonsei.ac.kr).

Digital Object Identifier 10.1109/TSM.2017.2676245

An ANN model consists of an input layer, multiple hidden layers, and an output layer. Each layer is composed of nodes, which are connected with the nodes of the succeeding layer. The strength of an internode connection is determined by a weight; the higher the weight, the greater the extent to which the information of the preceding node is reflected in the succeeding node. As a result of dedicated efforts to improve weight learning methods of ANNs, critical low-performance problems were solved in the 2000s, caused by gradient vanishing occurring when the number of hidden layers increases, overfitting, and computing power constraints [6]. The efforts enabled the efficient training of neural networks with deep hidden-layer architectures. Current neural network models with deep architectures have the following distinctive features: structural optimization accomplished by varying the number and type of hidden layers, advanced learning methods to improve the classification performance when addressing more complex data, and functional evolution from a simple classifier role of a conventional ANN paradigm to an automatic feature extractor for complex classification problems.

Among currently available deep learning models, CNN takes center stage in the field of image recognition due to its high classification performance as an algorithm expressing the biological visual structure as a neural network, inspired by a cat's visual cortex [5]. Today, CNNs are also employed in the field of process monitoring for various manufacturing systems due to its capacity to automatically extract features. Lee *et al.* [7] achieved high classification accuracy in bearing fault detection using CNN on a signal dataset consisting of univariate and bivariate time series. Chen *et al.* [8] conducted a fault pattern diagnosis of gearboxes using the statistical measures and spectral information collected from time and frequency domains as the CNN input data. Zheng *et al.* [9] proposed a CNN application in which a multivariate time-series dataset is decomposed into univariate time-series data for each variable, and feature extraction was conducted for each variable in parallel, followed by a classification task with a multilayer perceptron (MLP) with the extracted features.

These studies are different from our study in that the studies did not extract features from multivariate time-series data while considering the co-relationship among a plurality of variables. In semiconductor manufacturing processes, all recipe parameters should reach their individual set points in a timely manner and maintain the set points without severe fluctuations for specified process durations. More importantly, the state of each parameter should evolve with those of other parameters by preserving the co-behavior relationship as indicated in the recipe.

The CNN proposed in this study (hereinafter "FDC-CNN") is designed to deliver high performance on FDC tasks with multivariate time-series data. The main advantage of FDC-CNN is its feature extraction method, in which inter-variable correlations are analyzed by representing the input data in the form of a two-dimensional matrix with the variable axis and the time axis and finding the correlations along the time axis. Moreover, the proposed model provides information on the variables crucial for fault diagnosis, enabling the identification of differences between normal and faulty patterns by plotting the raw signals of the identified variables. In a comparison experiment, FDC-CNN outperformed other deep learning models in terms of training speed and classification accuracy.

The remainder of this paper is organized as follows. Section II introduces representative deep learning models, and Section III presents the proposed FDC-CNN. Section IV describes a performance evaluation experiment with chemical vapor deposition (CVD) process data, in which the training speed and classification performance of FDC-CNN and other deep learning models were compared, and presents the fault diagnosis results obtained by FDC-CNN. Finally, Section V concludes this study and discusses future research topics.

II. Deep Learning Models

This section introduces deep learning models widely used in the field of FDC. First, the working mechanism of a standard CNN, which underlies the proposed algorithm, is explained in detail, followed by a brief introduction of the stacked autoencoder (SAE) and stacked denoising autoencoder (SdA) [10].

A. Standard Convolutional Neural Network

A CNN is composed of two parts, as illustrated in Fig. 1. One part is responsible for feature extraction and contains an input layer, convolutional layers, and pooling layers. Convolutional layers and pooling layers, which are stacked layer by layer in the network, are feature extractors. The other part is responsible for classification and contains fully connected layers and an output layer. In this part, fully connected layers receive the features obtained by the last pooling layer as the input and perform classification tasks.

The input layer receives data or images to be classified, and then, the convolutional layer detects the local features of the input and stores them as a feature map. As illustrated in Fig. 1, a connection between the input layer and the convolutional layer is established by the receptive field. A receptive field is a square matrix of weights whose size is substantially smaller than the input. A feature map contains nodes, each of which is connected to a certain input area defined by the receptive field. The receptive field strides across the input area along the horizontal and vertical axes and executes the convolution operation as expressed by (1).

$$y_{ij} = \sigma\left(\sum_{r=1}^{F}\sum_{c=1}^{F} w_{rc} x_{(r+i\times S)(c+j\times S)} + b\right),$$
$$0 \le i \le \frac{H-F}{S}, 0 \le j \le \frac{W-F}{S}. \quad (1)$$

In (1), y_{ij} denotes the output value of a node on the feature map; H and W stand for height and width, denoting the vertical and horizontal dimensions of the input data; F denotes the height and width size of the receptive field; and S stands for the stride length, denoting the step size of the receptive field's movement. The term $x_{(r+i\times S)(c+j\times S)}$ represents the input data element having the coordinate $(r+i\times S,\ c+j\times S)$, and w_{rc}

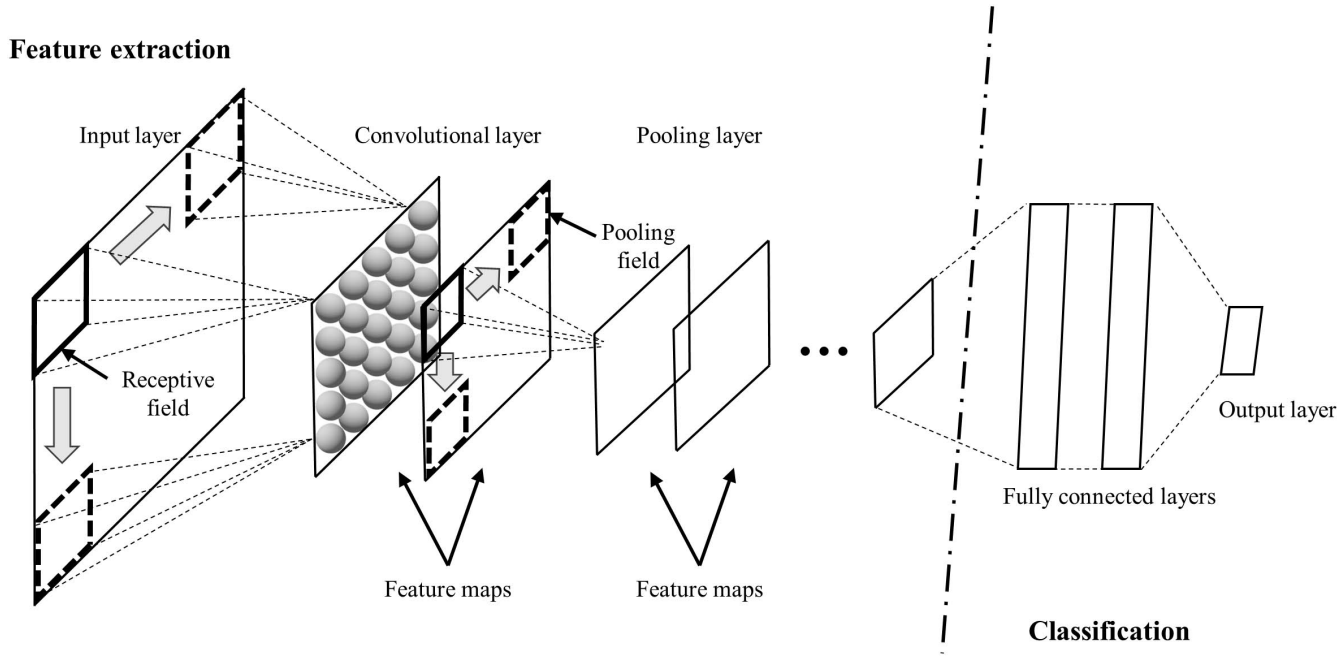

Fig. 1. Structure of convolutional neural network.

and b represent the weight positioned at (r, c) on the receptive field and the bias, respectively. The term σ denotes any nonlinear activation function to extract features of the input, with rectified linear unit (ReLU), hyperbolic tangent, and the sigmoid function being the most commonly used. In the CNN architecture, all nodes of a feature map share the same weights. In other words, the receptive field applied to create a feature map has a single weight matrix. This implies that the receptive field attempts to find a feature of a similar characteristic (*e.g.*, a single inter-variable correlation in the multivariate sensor signals) across the entire input area. In a convolutional layer, multiple feature maps are usually created to find different features by applying the receptive field iteratively with different weight matrices. In computing with (1), the input size $(H \times W)$ is reduced to the size of $(\frac{H-F}{S}+1) \times (\frac{W-F}{S}+1)$ for the convolutional layer, resulting in a gradual decrease in dimension as the stack of convolutional layers becomes deeper. To maintain the outcome of the convolution operation at the same dimension as that of the input, a padding of size $\frac{F-1}{2}$ is usually added at both edges of the input, and the stride length S is set to one. The pooling layer is usually located behind the convolutional layer and plays the role of reducing the size of the feature map using the pooling field and simultaneously constructs a condensed feature map by selecting location-invariant features. Max-pooling is the method most commonly used to reduce the map size. However, other methods, *e.g.*, average-pooling and L2-norm-pooling, are also used.

In Fig. 1, information of the condensed feature maps in the last pooling layer is transferred to the classification part. In this part, the nodes of the first hidden MLP layer are connected to each of the nodes constituting the condensed feature map. Finally, the output layer, which consists of the same number of nodes as there are classes to be classified, produces the probability score of each class. Specifically, the model executes the classification toward the side having a higher class score. Additionally, dropout [11] can be used in the classification part to improve the generalization performance of the model. Dropout skips the weight update of certain hidden nodes at each training round. It is known to be effective for preventing overfitting of networks to training data.

B. Stacked (Denoising) Autoencoder

An autoencoder (AE) is a neural network model that reproduces input data at the output layer with a small number of hidden nodes. The encoder, which corresponds to input and hidden layers, represents input data in a reduced dimension defined by the number of hidden nodes. The decoder, which indicates hidden and output layers, recovers the input data abstracted in the hidden layer by the encoder to the output nodes. The training of an AE determines the weights of the hidden layer such that output results are identical to input data to the greatest extent possible.

A SAE that stacks AEs can extract highly complex features by using nonlinear activation functions. SAE employs a layer-by-layer training method, which learns hidden layers one by one. For example, in the case of an SAE with two hidden layers, the weights between the input layer and the first hidden layer are trained first; next, the weights that are positioned between the first and second hidden layers are trained. After all hidden layers have been sequentially pretrained, a final fine tuning is conducted. Fine tuning is a simultaneous retraining of all weights in the entire network with class label information and makes SAE a high-performance classifier.

A SdA conducts the pretraining stage using a denoising autoencoder (dA) as the basic training unit instead of an AE. The basic architecture of dA is identical to that of an AE. It utilizes a different training method in that noised

data are input into its input layer, and the hidden layer is trained such that the output layer produces the original data. Thus, a dA generates a neural network that is more robust to corrupted input data than is the AE model. A SdA, which is built by stacking dAs, conducts layer-by-layer pretraining followed by fine tuning, as an SAE does.

III. Proposed Method

A. FDC-CNN Model

A sensor signal dataset collected in the course of a wafer fabrication process can be represented as a two-dimensional matrix with a processing time axis and a sensor variable axis. A feature map can be extracted using a conventional square receptive field. However, it is not desirable to apply this approach for the wafer data classification because the square receptive field, with a size of $(F\times F)$, cannot extract correlations among all sensor variables. A salient property of FDC-CNN is the configuration of its receptive field being tailored to the architecture of multivariate time-series data to enable the receptive field to move along the time axis only, therein extracting local inter-variable correlation features. Fig. 2 shows the architecture of the convolutional layer of the proposed model. Suppose that the sensor signals of a wafer consist of measurements collected from J sensor variables during K time intervals. FDC-CNN receives the sensor signals via the input layer composed of $K \times J$ nodes. In addition, FDC-CNN sets J as the column size of the receptive field. Accordingly, the output of a node in the feature map is defined by

$$y_i=\sigma\left(\sum_{r=1}^{F_r}\sum_{c=1}^{J} w_{rc}x_{(r+i\times S_r)(c)}+b\right), 0 \le i \le \frac{K-F_r}{S_r}, \quad (2)$$

where F_r and S_r denote the row size and stride length of the receptive field, respectively. Because, in FDC-CNN, the receptive field extracts local features while moving along the processing time axis only, a one-dimensional feature map is generated, unlike the two-dimensional structure of the standard CNN feature map. Subsequently, dimensional reduction is conducted in the pooling layer on the time axis only for each feature map, using the $(P\times 1)$ pooling field as illustrated in Fig. 2. As the number of stacked convolutional and pooling layers increases, two effects can be induced: 1) extracted features can reflect nonlinear inter-variable correlation on the convolutional layer side, and 2) more distinct features, which are significant for classification, are automatically learned due to the decreased resolution of sensor signals on the pooling layer side. The classifier consists of fully connected layers and an output layer, using the feature maps of the last pooling layer as input, as in the standard CNN.

B. Fault Diagnosis

In FDC-CNN, the variable and time information crucial for fault classification can be identified using the weights of the receptive field and the feature map in the first convolutional layer. During the training phase of the CNN, weights

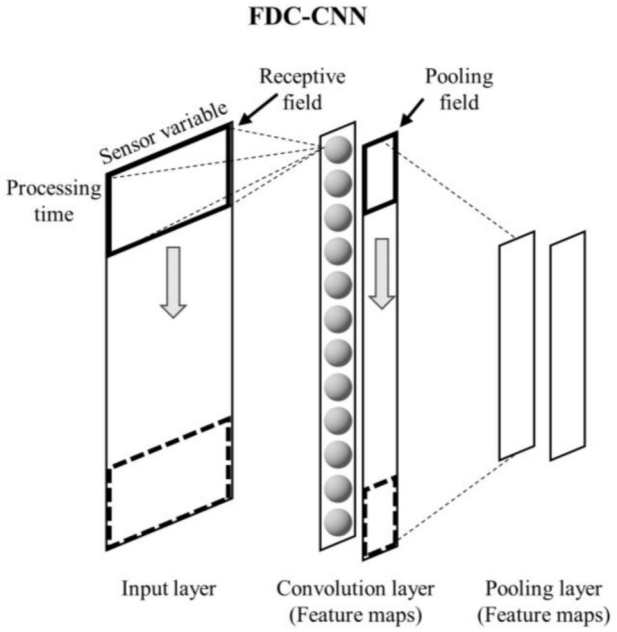

Fig. 2. Feature extraction in FDC-CNN.

are updated using the gradient descent method to minimize the loss (classification error) function value. This implies that the larger the magnitude of a given weight, the greater the extent to which the preceding nodes contribute to extracting features significant for classification. In the case of FDC-CNN, each weight column of the receptive field of the first convolutional layer represents the importance of a specific sensor variable. Therefore, the contribution level of variable (CLV) of sensor variable j can be determined as follows:

$$\mathrm{CLV}_j = 100\times\left(d_j - \sqrt{J}\ \mathrm{median}(\boldsymbol{d})\right)^+, \quad (3)$$

where

$$\overline{w_j} = \frac{\sum_{r=1}^{F_r}\left(w_{rj}\right)^+}{F_r} \text{ and } d_j = \left|\overline{w_j} - \ \mathrm{median}\ (\overline{\boldsymbol{w}})\right|.$$

The $\overline{w_j}$ term represents the mean of the positive weights of the j-th receptive field column (*i.e.*, sensor variable j); the $\overline{\boldsymbol{w}}$ term is the set of $\overline{w_1}, \ldots, \overline{w_J}$; the d_j term represents the difference between the $\overline{w_j}$ and the median of $\overline{w_1}, \ldots, \overline{w_J}$; and the $\boldsymbol{d}$ term is the set of $d_1, \ldots, d_J$. CLV is a variable selection measure based on the Hampel's outlier detector [12]. Hampel suggested the median of absolute deviation as robust estimates for outlier identification. In (3), the measure assigns a high score to the variable j when the d_j (absolute deviation from the mean weight) is bigger than the median of $d_1, \ldots, d_J$, considering the variable j as an important variable in forming the feature map. This approach does not require any statistical assumption for the mean weight distribution. In (3), $\sqrt{J}$ controls the number of selected variables and suppresses an increase in the number of variables with high CLVs as the number of sensor variables increases. In FDC-CNN, if there is only one sensor variable having a positive CLV in a receptive field, the receptive field selects the variable as a single feature for fault classification. If multiple sensor variables have positive CLVs

in another receptive field, the receptive field detects the correlation of the variable with positive CLVs as an effective feature of fault classification. If there is a receptive field in which all variables have negative CLVs, this can be regarded as a failure to find meaningful local fault features with the receptive field.

The feature map of the first convolutional layer has the same row (processing time) size as the input data, and node i of the feature map stores the activation degree y_i of a local input feature through the convolution between weights of the receptive field and connected input data as in (2). Thus, each element y_i reflects the state of the connected input data. Consequently, a trained feature map with a minimized total classification error shows similar activation patterns across the processing time axis for data of the same class labels and heterogeneous patterns for data of different class labels. To determine the time sections of faulty wafer data displaying local features (patterns) that differ from normal wafer data, the feature map of a wafer is normalized as follows:

$$z_i = g\left(\frac{y_i - \overline{y_i^{\text{normal}}}}{\sqrt{\text{var}\left(y_i^{\text{normal}}\right)}}\right),\ 1 \le i \le K,$$
$$g(x; a) = \begin{cases} \text{minimum } (x,\ a), & \text{if } x \ge 0, \\ \text{maximum } (x, -a), & \text{if } x < 0, \end{cases} \tag{4}$$

where z_i denotes the i-th element of the feature map of the wafer normalized using the mean and variance of the feature map of a normal class dataset. In the normalization process, outliers in the feature map are restricted by the function g to the range between the maximum value a and the minimum value $-a$. The degree of significance of the change in the local feature is demonstrated by the value of z generated for each wafer. Therefore, a normalized feature map showing an obvious difference between normal and fault classes provides information on the processing time section at which different patterns were displayed in the raw data.

To summarize, if wafers show similar distortion patterns in a normalized feature map during a specific processing time section, these peculiar changes can provide insight into the root cause of a process disturbance. Sensor variables with high CLV values may be suspected of having caused the disturbance. Therefore, sensor variables with high CLV values and the time sections showing different patterns in normalized feature map can be used for fault diagnosis.

IV. Experiment

A. Data and Setup

In this section, we describe our experiment based on data collected from a work-site CVD tool. The data used are sensor measurements collected at 0.2 second intervals for approximately 105 seconds during the manufacturing process for ten sensor variables, including temperature, pressure, and gas flow rate for each wafer. In data preprocessing, the sensor measurements were scaled to a range of between 0 and 1. There are six wafer classes, consisting of normal wafer and five types of faulty wafers defined by on-site workers. A CVD simulator was constructed using the real data to train and test FDC-CNN. The training set is composed of 5,000 normal data samples and 5,000 fault data samples (1,000 for each fault type), and the test set consisted of 1,000 normal data samples and 1,000 fault data samples (200 for each fault type).

The following deep learning models were tested in the experiment: FDC- CNN, standard CNN, SAE, SdA, and ANN. For FDC-CNN and standard CNN, we stacked three pairs of convolutional and pooling layers with 10, 20, and 40 feature maps and performed feature extraction. For the classifier, we used two fully connected layers with 500 and 100 hidden nodes. For FDC-CNN, we used (3×10) receptive fields in the first convolutional layer and (3×1) receptive fields in the succeeding layers, and receptive fields in the convolutional layer of standard CNN were set to 3×3. The stride length was set to one for all models. Max-pooling was used in the pooling layer, with the sizes of the pooling fields for standard CNN and FDC-CNN set at (2×2) and (2×1), respectively. Because standard CNN can achieve different performance levels depending on whether dropout is applied, both cases were considered in the experiment.

The fully connected layer-based models, SAE, SdA, and ANN, were given three hidden layers with 5,000, 1,000, and 200 nodes, respectively. In the cases of SAE and SdA, layer-wise pretraining was performed on each of the three hidden layers, followed by fine tuning. The ReLU function was applied as the activation function to all the deep learning models, and the softmax cross-entropy loss function was employed for model training. In addition, SVM, which is a high-performance classifier, was considered. Specifically, two SVMs with a linear kernel function (SVM.L) and a radial basis function (SVM.R) were included in the performance evaluation. The cost parameter was optimized to 1 for SVM.L, and the gamma and cost parameters were optimized to 0.1 and 10, respectively, for SVM.R.

B. Training Parameters and Speed

Among the major parameters to be selected in the training of deep learning models, the learning rate and batch size were optimized in a heuristic manner. For the learning rate, which represents the rate for updating the connection weights, we used values ranging from 10^{-5} to 10^{-1}, incrementing the exponent by 0.5. The batch size is the amount of training data used for one weight update; the lower the value, the more frequently the weight is updated and the more time is required for model training. Considering the size of the training dataset (10,000 data samples), the batch sizes were set at 100, 40, and 20 to induce weight update frequencies of 100, 250, and 500 per training epoch, and we checked the model training performance for each epoch. Here, one epoch corresponds to one pass through the entire set of training data. We trained each model for 50 epochs. The results confirmed that the performance of FDC-CNN and standard CNN reached their peaks at a learning rate of 10^{-3}, and the peak rates of ANN, SAE, and SdA were obtained at 10^{-4}. As for batch size, all the deep learning models achieved their highest performance with a size of 20 data samples.

The graphs in Fig. 3 represent the training speeds of the comparison models with the optimized parameter values.

TABLE I
FAULT CLASSIFICATION PERFORMANCE

Method	Overall accuracy	Normal	Fault A	Fault B	Fault C	Fault D	Fault E	Training time (s)
FDC-CNN	**97.9%**	99.3%	**99.2%**	93.0%	**100.0%**	97.8%	**92.4%**	338.3
Standard CNN	97.0%	99.0%	96.9%	94.9%	99.9%	97.2%	85.6%	747.5
Standard CNN_drop	97.1%	99.0%	98.9%	**96.3%**	**100.0%**	**98.2%**	82.4%	887.0
SAE	91.8%	99.1%	95.7%	84.4%	97.4%	82.2%	62.8%	1159.5
SdA	90.2%	97.9%	94.8%	85.1%	97.9%	78.5%	55.5%	1221.0
ANN	81.2%	88.5%	96.5%	74.8%	97.0%	50.8%	50.3%	960.5
SVM.L	76.2%	99.8%	44.5%	42.5%	54.5%	47.5%	73.5%	1390.1
SVM.R	81.5%	98.7%	87.5%	51.0%	77.5%	49.0%	56.5%	1357.8

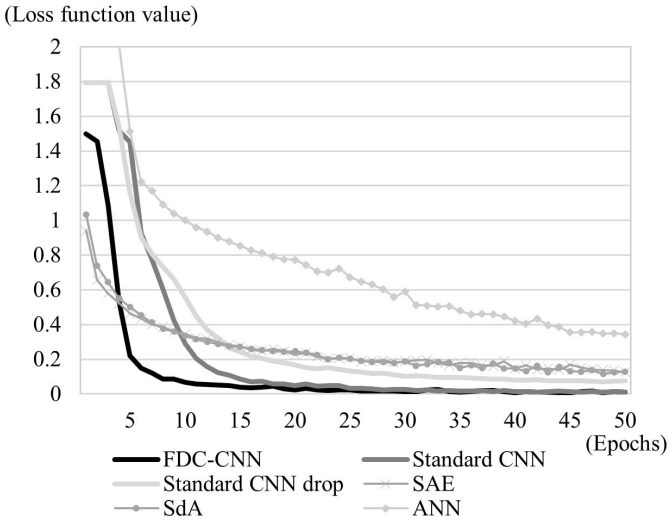

Fig. 3. Comparison of training speeds.

In the graphs, the number of epochs is on the x-axis, and the loss function value is on the y-axis. The loss function value is an indicator of the training quality of a model: the faster the value decreases, the higher the training speed of the neural network model. Among the deep learning models compared in this study, FDC-CNN demonstrated the highest training speed, marking a cost of lower than 0.1 after 10 epochs. The standard CNN model with dropout showed cost value stagnation for approximately 4–5 epochs at an early stage of training and was slightly outpaced by FDC-CNN. The training speeds of the SAE and SdA models were recorded only during the fine-tuning process. Both models showed patterns of rapid training progress at early stages due to the proper weight initialization performed during pretraining but then slowed as the training advanced. This may be ascribable to the training load of a larger number of weights compared with the CNN-based approach in which weight sharing is applied. Finally, the traditional ANN model was confirmed to have a lower learning speed than the SAE and SdA models because training started using random initial weights without proper pretraining.

C. Fault Classification Results

Table I presents the results of the performance comparison between FDC-CNN and the other models. Model training was conducted using the optimized parameter values obtained in the preliminary experiment. First, the models with the CNN-based approach demonstrated over 99% classification accuracy on the normal wafers and achieved the best performance; of these, FDC-CNN was found to have the highest overall accuracy score (97.9%). Regarding fault classification, FDC-CNN and the standard CNN with dropout outperformed the other models in classifying three of the five fault types. In the case of fault type E, which was the most difficult fault type to detect, FDC-CNN outperformed the second-best model (standard CNN) by 6.8%. Regarding the aspect of learning time, FDC-CNN obtained the shortest training time among the CNN-based models, thus achieving the highest training speed of all the algorithms because it used the largest receptive field and thus required only a small number of convolution operations. Among the fully connected models, it was also confirmed that models conducting pretraining in the course of neural network training, such as SAE and SdA, showed overall accuracy improvements of 10.6% and 9.0%, respectively, over that of ANN, although they required longer training times. However, these fully connected models achieved lower performances than did the CNN-based models in terms of both classification accuracy and training time. Compared with the deep learning models, the SVM models performed at similar levels in terms of classification accuracy on the normal wafers but showed relatively low performance in classifying fault types A, B, and C.

D. Fault Diagnosis Results

FDC-CNN has the advantage of enabling an intuitive understanding of the model's decision making for fault diagnosis using information in the first convolutional layer. Fig. 4 shows the normalized feature maps with $a = 5$, and each fault showing the difference pattern against the normal wafers is visualized based on this information. In addition, CLV values of the major sensor variables were derived using the weight matrix of the receptive field, and Fig. 4(a) illustrates the result of the fault diagnosis for fault type A. By comparing the normalized feature maps of the normal and faulty wafers, it can be confirmed that a process disturbance occurred in the faulty wafer signal at the approximately 400–430 time intervals, whereas the normal wafer produced a clean image. The corresponding CLVs were presented in the order of 4.6 (sensor 10), 1.4 (sensor 4), and 1.0 (sensor 7), and thus, these sensor variables were found to be the major source of the fault type. The signal plots of normal wafers (black line)

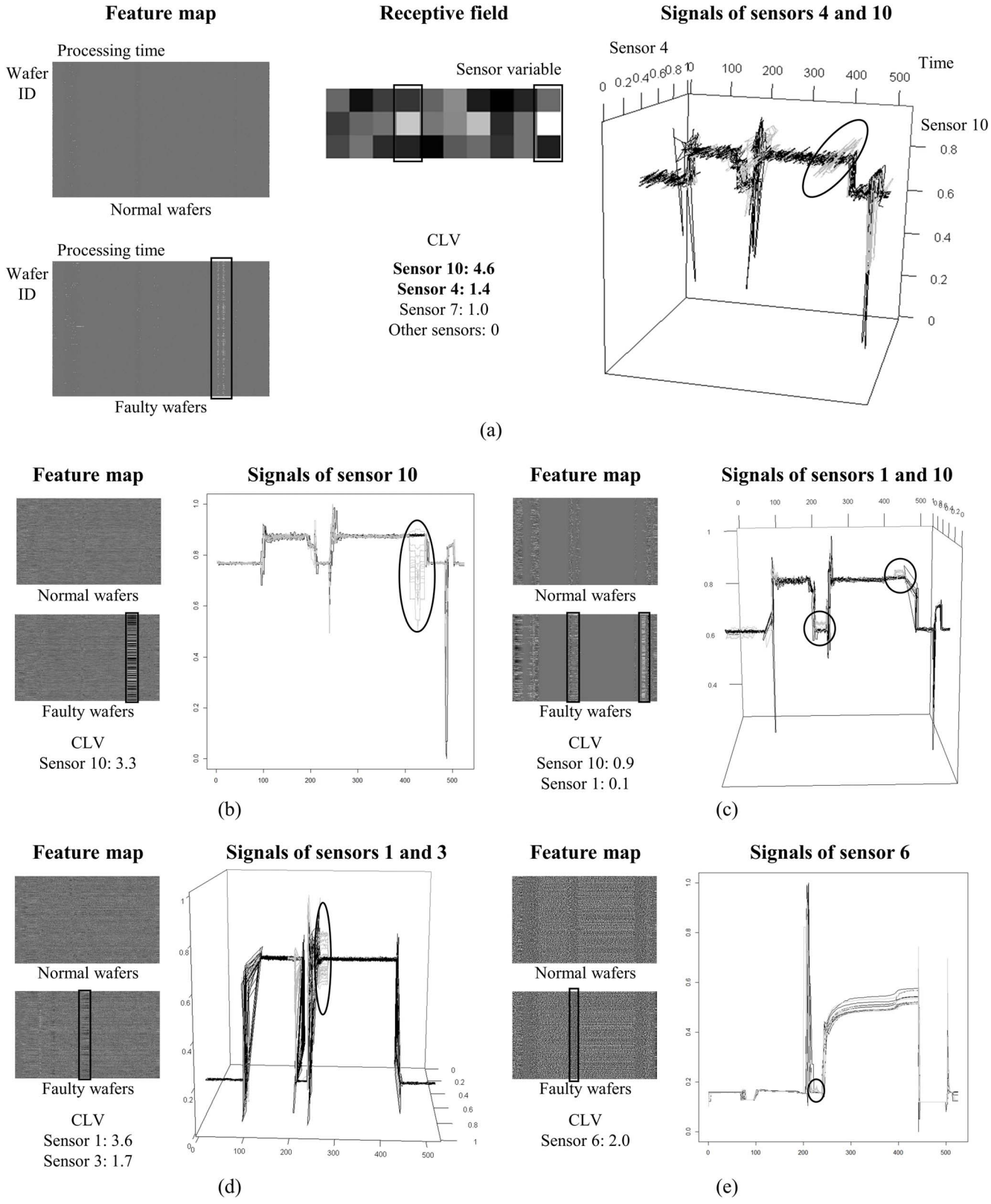

Fig. 4. Examples of FDC-CNN-based fault diagnosis: feature maps denote normalized ones; signal plotting for normal wafer (black line) and fault wafer (gray line); (a) Fault type A, (b) Fault type B, (c) Fault type C, (d) Fault type D, (e) Fault type E.

and faulty wafers (gray line) for sensor variables 10 and 4 (with high CLV values) clearly show pattern differences in the time section where a disturbance occurred on the normalized feature maps. Fig. 4(b)–(e) presents the fault diagnosis results for fault types B, C, D, and E in the same manner, wherein two-dimensional (variable axis and time axis) signal plotting

was performed in cases of one variable having a high CLV value and three-dimensional signal plotting in cases of two such variables. This accentuates the clear pattern differences between normal and faulty wafers in the normalized feature maps within the corresponding time section.

Such interpretations are possible because the FDC-CNN model's convolutional layer was designed considering the multivariate time-series data so that the outputs of the convolutional layer provide the fault-related structural meaning of the raw data. In FDC-CNN, not only is the receptive field designed to enable a simultaneous consideration of all inter-variable correlations, but it can also be used to measure the contribution level of each sensor variable to the fault classification due to the one-to-one matching between weight columns and sensor variables. Furthermore, the normalized feature maps generated by the proposed method are matched to the processing time axis of the raw data in a manner that provides fault occurrence time information during the manufacturing process and thus can be used for efficient process monitoring.

V. Conclusion

In this study, CNN-based fault classification and fault diagnosis were conducted on multivariate sensor signals for semiconductor process monitoring. In the proposed FDC-CNN model, the convolutional layer of standard CNN was redesigned to consider the data's structural characteristics, leading to an improvement in training speed and classification performance. In particular, FDC-CNN provides major sensor variables and time section information to enable mapping to the raw data, thus providing insights that are useful for fault diagnosis without requiring empirical or specialized knowledge.

The following research issues may be considered for follow-up studies. First, the fault diagnosis presented in the proposed model uses only the information in the first convolutional layer. Information contained in deep convolutional layers has a limited potential for analyzing inter-variable correlations because features maps extracted in the deep convolutional layers are the results of highly nonlinear functions of the raw data and thus are difficult to interpret. However, de-convolutional neural networks [13], which are being extensively studied in the field of image classification as a method for assigning pixel-wise class labels to raw image pixels using switch variables, can be applied to fault classification; information contained in deep convolutional layers could be used for fault diagnosis. Second, there is a pressing need for an FDC model that explains fault types not encountered during the training stage that can occur because of increasingly more complicated manufacturing process steps. In response to this need, FDC research will be required to correctly classify—as a new class—any novel type of fault encountered in the field.

References

[1] J. J. Gertler, *Fault Detection and Diagnosis in Engineering Systems*. Boca Raton, FL, USA: CRC Press, 1998.

[2] J. M. Nawaz, M. Z. Arshad, and S. J. Hong, "Fault diagnosis in semiconductor etch equipment using Bayesian networks," *J. Semicond. Technol. Sci.*, vol. 14, no. 2, pp. 252–261, Apr. 2014.

[3] C.-F. Chien, C.-Y. Hsu, and P.-N. Chen, "Semiconductor fault detection and classification for yield enhancement and manufacturing intelligence," *Flexible Serv. Manuf. J.*, vol. 25, no. 3, pp. 367–388, Sep. 2013.

[4] S. Mahadevan and S. L. Shah, "Fault detection and diagnosis in process data using one-class support vector machines," *J. Process Control.*, vol. 19, no. 10, pp. 1627–1639, Dec. 2009.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 26th Annu. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, 2012, pp. 1106–1114.

[6] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[7] D. Lee, V. Siu, R. Cruz, and C. Yetman, "Convolutional neural net and bearing fault analysis," in *Proc. Int. Conf. Data Min.*, Las Vegas, NV, USA, 2016, pp. 194–200.

[8] Z. Chen, C. Li, and R.-V. Sanchez, "Gearbox fault identification and classification with convolutional neural networks," *Shock Vib.*, vol. 2015, Apr. 2015, Art. no. 390134.

[9] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *Proc. 15th Int. Conf. Web-Age Inf. Manag.*, Macau, China, 2014, pp. 298–310.

[10] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.

[11] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.

[12] L. Davies and U. Gather, "The identification of multiple outliers," *J. Amer. Stat. Assoc.*, vol. 88, no. 423, pp. 782–792, Sep. 1993.

[13] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. Int. Conf. Comput. Vision*, Santiago, Chile, 2015, pp. 1520–1528.

Ki Bum Lee received the B.S. degree in industrial engineering from Yonsei University, South Korea, in 2014, where he is currently pursuing the Ph.D. degree in industrial engineering. His current research interest is machine learning for manufacturing.

Sejune Cheon received the B.S. degree in industrial engineering from Ajou University, South Korea, in 2011, where he is currently pursuing the Ph.D. degree in industrial engineering. His current research interest is data science for manufacturing industries.

Chang Ouk Kim received the B.S. and M.S. degrees from Korea University, Seoul, South Korea, in 1988 and 1990, respectively, and the Ph.D. degree from Purdue University, West Lafayette, IN, USA, in 1996, all in industrial engineering. He is a Professor with the Department of Industrial Engineering, Yonsei University, South Korea. He has published over 100 papers in journals and conference proceedings. His current research interest is data science for manufacturing.