International Journal of Internet, Broadcasting and Communication Vol.9 No.2 20-26 (2017)
https://doi.org/10.7236/IJIBC.2017.9.2.20

IJIBC 17-2-3

Real-time Fault Detection in Semiconductor Manufacturing Process : Research with Jade Solution Company

Byung Joo Kim

Department of Computer Engineering, Youngsan University, Korea
bjkim@ysu.ac.kr

Abstract

Process control is crucial in many industries, especially in semiconductor manufacturing. In such large-volume multistage manufacturing systems, a product has to go through a very large number of processing steps with reentrant) before being completed. This manufacturing system has many machines of different types for processing a high mix of products. Each process step has specific quality standards and most of them have nonlinear dynamics due to physical and/or chemical reactions. Moreover, many of the processing steps suffer from drift or disturbance. To assure high stability and yield, on-line quality monitoring of the wafers is required. In this paper we develop a real-time fault detection system on semiconductor manufacturing process. Proposed system is superior to other incremental fault detection system and shows similar performance compared to batch way.

Keywords: *Semiconductor manufacturing process, Conjugate least squares support vector machine, real-time*

1. Introduction

Fault detection in semiconductor manufacturing process is a hot research topic as more and more sensor data are being collected throughout the industrial process. Many machine learning algorithms used in pattern classification are now being utilized in fault detection. Dimension reduction techniques, such as principal component analysis, partial least squares, and Fisher's discriminant analysis have been applied to detect faults in chemical processes[1][2]. Support vector machine and artificial neural networks are also widely used methods for fault detection; they have been applied to gearbox failure detection[3] and chemical process fault diagnosis[4]. K-Nearest Neighbor and fuzzy-logic are two other powerful methods that have been used to detect faults in semiconductor manufacturing processes[5] and mechanical systems [6]. Tree based algorithms such as random forest and gradient boosted tree are useful machine learning algorithms in situations where one expects nonlinear and interactive effects between covariates. They have been applied to fault detection in aircraft system[7]. Recently kernel trick has been applied to PCA and is based on a formulation of PCA in terms of the dot product matrix instead of the covariance matrix[8]. Kernel

Manuscript Received: Mar. 12, 2017 / Revised: Mar. 20, 2017 / Accepted: Apr. 5, 2017
Corresponding Author: bjkim@ysu.ac.kr
Tel: +82-055-380-9447, Fax: +82-055-380-9447
Department of Computer Engineering, Youngsan University

PCA(KPCA), however, requires storing and finding the eigenvectors of a $N \times N$ kernel matrix where N is a number of patterns. It is infeasible method for when N is large. This fact has motivated the development of empirical kernel method which does not store the kernel matrix. In this paper we propose a method that allows for incremental eigenspace update method by incremental kernel PCA for detecting the wafer fault. Paper is organized as follows. In Section 2 we will briefly explain the incremental PCA method. In Section 3 KPCA and conjugate least squares support vector machine(LS-SVM) are introduced and to make KPCA incrementally, empirical kernel map method is explained. Experimental results to evaluate the performance of proposed method is shown in Section 4. Discussion of proposed method and future work is described in Section 5.

2. Incremental PCA

In this section, we will give a brief introduction to the method of incremental PCA algorithm which overcomes the computational complexity of standard PCA. Before continuing, a note on notation is in order. Vectors are columns, and the size of a vector, or matrix, where it is important, is denoted with subscripts. Particular column vectors within a matrix are denoted with a superscript, while a superscript on a vector denotes a particular observation from a set of observations, so we treat observations as column vectors of a matrix. As an example,

A_{mn}^{i} is the ith column vector in a $m \times n$ matrix. We denote a column extension to a matrix using square brackets. Thus $[A_{mn} b]$ is an(m × (n + 1)) matrix, with vector b appended to A_{mn} as a last column.

To explain the incremental PCA, we assume that we have already built a set of eigenvectors $U = [u_j, j = 1, \cdots, k]$ after having trained the input images $x_i, i =, \cdots, N$. The corresponding eigenvalues are Λ and $\overline{X}$ is the mean of input image. Incremental building of eigenspace requires updating these eigenspace to take into account of a new input image. Here we give a brief summarization of the method which is described in [9]. First, we update the mean:

$$\overline{x}' = \frac{1}{N+1}(N\overline{x} + x_{N+1}) \tag{1}$$

We then update the set of eigenvectors to reflect the new input image and to apply a rotational transformation to U . For doing this, it is necessary to compute the orthogonal residual vector $\hat{h} = (Ua_{N+1} + \overline{x}) - x_{N+1}$ where

a_{N+1} is principal component and normalize it to obtain $h_{N+1} = \frac{h_{N+1}}{\|h_{N+1}\|_2}$ for $\|h_{N+1}\|_2 \rangle 0$ and $h_{N+1} = 0$ otherwise. We obtain the new matrix of eigenvectors U' by appending h_{N+1} to the eigenvectors U and rotating them :

$$U' = [U, h_{N+1}]R \tag{2}$$

where $R \in \Re_{(k+1)\times(k+1)}$ is a rotation matrix. R is the solution of the eigenspace of the following form:

$$DR = R\Lambda' \tag{3}$$

where Λ' is a diagonal matrix of new eigenvalues. We compose D $\in \Re_{(k+1)\times(k+1)}$ as:

$$D = \frac{N}{N+1}\begin{bmatrix} \Lambda & 0 \\ 0^T & 0 \end{bmatrix} + \frac{N}{(N+1)^2}\begin{bmatrix} aa^T & \gamma a \\ \gamma a^T & \gamma^2 \end{bmatrix} \tag{4}$$

where $\gamma = h_{N+1}^T(x_{N+1} - \bar{x})$ and $a = U^T(x_{N+1} - \bar{x})$. Though there are other ways to construct matrix D [8][9], the only method ,however, described in [9] allows for the updating of mean.

2.1 Updating Image Representations

The incremental PCA represents the input image with principal components $a_{i(N)}$ and it can be approximated as follows:

$$x_{i(N)} = \hat{U}a_{i(N)} + \bar{x} \tag{5}$$

To update the principal components $a_{i(N)}$ for a new image x_{N+1}, computing an auxiliary vector η is necessary. η is calculated as follows:

$$\eta = \left[U \; \hat{h}_{N+1}\right]^T (\bar{x} - \bar{x}') \tag{6}$$

then the computation of all principal components is

$$a_{i(N+1)}(R')^T \begin{bmatrix} a_{i(N)} \\ 0 \end{bmatrix} + \eta, \quad i = 1, \cdots, N+1 \tag{7}$$

The transformations described above yield a model that represents the input images with the same accuracy as the previous one, therefore we can now discard the old subspace and the coefficients that represent the image in it. x_{N+1} is represented accurately as well, so we can safely discard it. The representation of all N + 1 images are possible because the subspace is spanned by k +1eigenvector. Due to the increase of the dimensionality by one, however, more storage is required to represent the data. If we try to keep a k-dimensional eigenspace, we lose a certain amount of information. In order to balance the storage requirements with the level of accuracy, it is needed for us to set the criterion on retaining the number of eigenvectors. There is no explicit guideline for retaining a number of eigenvectors.

In this paper we set our criterion on adding an eigenvector as $\lambda'_{K+1} \rangle 0.7\bar{\lambda}$ where $\bar{\lambda}$ is a mean of the λ. Based on this rule, we decide whether adding u'_{K+1} or not.

2.2 Empirical Feature Map

A prerequisite of the incremental eigenspace update method is that it has to be applied on the data set. Furthermore incremental PCA builds the subspace of eigenvectors incrementally, it is restricted to apply the linear data. But in the case of KPCA this data set $\Phi(x^N)$ is high dimensional and most of the time can not even be calculated explicit ly. For the case of nonlinear data set, applying feature mapping function method to incremental PCA may be one of the solutions. This is performed by so-called kernel-trick , which means an implicit embedding to an infinite dimensional Hilbert space[8] (i.e. feature space) F .

$$\mathrm{K}(x, y) = \Phi(x) \cdot \Phi(y) \tag{8}$$

Where K is a given kernel function in an input space. When K is semi positive definite, the existence of Φ is proven[8]. Most of the case, however, the mapping Φ is high-dimensional and cannot be obtained explicitly. The vector in the feature space is not observable and only the inner product between vectors can be observed

via a kernel function. However, for a given data set, it is possible to approximate Φ by empirical kernel map proposed by Scholkopf and Tsuda[8] which is defined as $\Psi_N : \Re^d \to \Re^N$

$$\Psi_N(x) = [\Phi(x_1)\cdot\Phi(x), \cdots, \Phi(x_N)\cdot\Phi(x)]^T = [K(x_1, x), \cdots, K(x_N, x)]^T \tag{9}$$

A performance evaluation of empirical kernel map was shown by Tsuda. He shows that support vector machine with an empirical kernel map is identical with the conventional kernel map[12]. The empirical kernel map $\Psi_N(x_N)$, however, do not form an orthonormal basis in $\Re^N$, the dot product in this space is not the ordinary dot product. In the case of KPCA , however, we can be ignored as the following argument. The idea is that we have to perform linear PCA on the $\Psi_N(x_N)$ from the empirical kernel map and thus diagonalize its covariance matrix. Let the N × N matrix $\Psi = [\Psi_N(x_1), \Psi_N(x_2), \cdots, \Psi_N(x_N)]$, then from equation (9) and definition of the kernel matrix we can construct Ψ = NK. The covariance matrix of the empirically mapped data is:

$$C_\Psi = \frac{1}{N}\Psi\Psi^T = NKK^T = NK^2 \tag{10}$$

In case of empirical kernel map, we diagonalize NK^2 instead of K as in KPCA. Mika shows that the two matrices have the same eigenvectors $\{u_k\}$ [12]. The eigenvalues $\{\lambda_k\}$ of K are related to the eigenvalues $\{K_k\}$ of NK^2 by

$$\lambda_k = \sqrt{\frac{K_k}{N}} \tag{11}$$

and as before we can normalize the eigenvectors $\{v_k\}$ for the covariance matrix C_Ψ of the data by dividing each $\{u_k\}$ by $\sqrt{\lambda_k N}$. Instead of actually diagonalize the covariance matrix C_Ψ , the incremental KPCA is applied directly on the mapped data Ψ = NK. This makes it easy for us to adapt the incremental eigenspace update method to KPCA such that it is also correctly takes into account the centering of the mapped data in an incremental way. By this result, we only need to apply the empirical map to one data point at a time and do not need to store the N × N kernel matrix.

3. Conjugate LS-SVM

Support vector machines(SVM) developed by Vapnik[8] and it is a powerful methodology for solving problems in nonlinear classification. Originally, it has been introduced within the context of statistical learning theory and structural risk minimization. In the methods one solves convex optimization problems, typically by quadratic programming(QP). Solving QP problem requires complicated computational effort and need more memory requirement. LS-SVM[10] overcomes this problem by solving a set of linear equations in the problem formulation. LS-SVM method is computationally attractive and easier to extend than SVM. But traditional batch way LS-SVM requires storing (N+1) × (N+1) matrix where N is a number of patterns. It is infeasible method when dealing with image data because its size is big. For image data sets the use of iterative methods is recommended. In principle, various methods can be used at this point including SOR(Successive Over-Relaxation), CG(Conjugate Gradient), GMRES(Generalized Minimal Residual) etc. However, not all of these iterative methods can be applied to any kind of linear system. For example, in order to apply CG the matrix should be positive definite. Due to the presence of the *b* bias term in the LS-SVM model the resulting matrix is not positive definite. So before we can apply such methods we have to transform the linear system into a positive definite system. The LS-SVM KKT system is of the form

$$\begin{bmatrix} 0 & Y^T \\ Y & H \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \quad (12)$$

more specifically with $H = \Omega + I/\gamma$, $\xi_1 = b, \xi_2 = \alpha$, $d_1 = 0, d_2 = I_v$. This can be transformed into

$$\begin{bmatrix} s & 0 \\ 0 & H \end{bmatrix} \begin{bmatrix} \xi_1 \\ \xi_2 + H^{-1} y \xi_1 \end{bmatrix} = \begin{bmatrix} -d_1 + y^T H^{-1} d_2 \\ d_2 \end{bmatrix} \quad (13)$$

with $S = y^T H^{-1} y > 0$ ($H = H^{-T} > 0$). Because *s* is positive and *H* positive definite the overall matrix is positive definite. This form is very suitable because different kinds of iterative methods can be applied to problems involving positive definite matrices. This leads to the LS-SVM classifier with conjugate gradient algorithm LS-SVM for big data is as follows.

1. Solve η, ν from $H\eta = Y$ and $H\nu = 1_v$
2. Compute $s = Y^T \eta$
3. Find solution
   $b = \eta^T 1_v / s$
   $\alpha = \nu - \eta b$

4. Experiment

To evaluate the performance of accuracy on eiegnspace update for incremental data, we take 1300 data and each data has 18 variables. Detailed variables attributes are shown in Table 1. Among data, the number of normal is 800 and rest of them is abnormal. Data were collected and recorded at one second intervals during the etch for each of these sensors. Since our primary concern in this work was to detect faults occurring from one wafer to the next, we took the average value of each variable during the etch process for each wafer, resulting in a 1x18 array of values for each wafer.

Table 1. Tool-state variables used for process monitoring

1	TCP Top Power	10	RF Impedance
2	TCP Tune	11	RF Power
3	TCP Load	12	TCP Reflected Power
4	TCP Phase Error	13	RF Bottom Reflected Power
5	TCP Impedance	14	Pressure
6	RF Bottom Power	15	BCI3 Flow
7	RF Tune	16	CI2 Flow
8	RF Load	17	He Pressure
9	RF Phase Error	18	Vat Value

4.1 Comparison with SVM

Recently SVM has been a powerful methodology for solving problems in nonlinear classification. To evaluate the classification accuracy of the proposed system it is desirable to compare with SVM. Generally a disadvantage of the incremental method is its accuracy compared to the batch method even though it has the advantage of memory efficiency. According to Table 2 and Table 3 we can see that the proposed method has better classification performance compared to batch SVM. Through this result we can show that the proposed classifier has remarkable classification accuracy, although it is worked in an incremental way.

Table 2. Performance comparison of proposed method and SVM using all features

	Training	Generalization	Eigenvalue update criterion
Standard SVM	100%	95.34%	none
Proposed method	100%	96.74%	$\lambda' \rangle 0.7\overline{\lambda}$

Table 3. Performance comparison of proposed method and SVM using extracted features

	Training	Generalization	Eigenvalue update criterion
Standard SVM	100%	95.02%	none
Proposed method	100%	97.03%	$\lambda' \rangle 0.7\overline{\lambda}$

5. Conclusion and Remarks

A conjugate based LS-SVM which combining empirical kernel map was presented for dealing with fault detection in semiconductor manufacturing process. Such classifier has following advantages. Proposed detection system is more efficient in memory requirement than batch LS-SVM. In batch LS-SVM the (N+1) × (N+1) matrix has to be stored, while for our proposed method does not. It is very useful when dealing with large size data. Experimental results on wafer data, proposed method shows lead to good performance. By this result we will make a commercial wafer fault detection system with Jade Solution Company.

Acknowledgement

This work was supported by a 2017 research grant from Youngsan University, Republic of Korea.

References

[1] L.H. Chiang, E.L. Russell, and R.D. Braatz, “Fault diagnosis in chemical processes using fisher discriminant analysis, discriminant partial least squares, and principal component analysis”, *Chemometrics and intelligent laboratory systems*, 50(2):243–252, 2000.

[2] L.H. Chiang, M.E. Kotanchek, and A.K. Kordon, “Fault diagnosis based on fisher discriminant analysis and support vector machines”, *Computers & chemical engineering*, 28(8):1389–1401, 2004.

[3] B. Samanta, “Gear fault detection using artificial neural networks and support vector machines with genetic algorithms”, *Mechanical Systems and Signal Processing*, 18(3):625–644, 2004.

[4] L. Wang, and J. Yu, “Fault feature selection based on modi- fied binary pso with mutation and its application in

chemical process fault diagnosis", *In Advances in Natural Computation*, pp. 832–840, Springer 2005.

[5] Q.P. He, and J. Wang, "Fault detection using the k-nearest neighbor rule for semiconductor manufacturing processes", *Semiconductor manufacturing, IEEE transactions on*, 20(4):345–354, 2007.

[6] J. Korbicz, J.M. Koscielny, Z. Kowalczuk, and W. Cholewa, "Fault diagnosis: models, artificial intelligence, applications", *Springer Science & Business Media*, 2012.

[7] S. Lee, W. Park, and S. Jung, "Fault detection of aircraft system with random forest algorithm and similarity measure", *The Scientific World Journal*, 2014.

[8] V.N. Vapnik, "Statistical learning theory", John Wiley & Sons, New York, 1998.

[9] J. Winkeler, B.S. Manjunath, and S. Chandrasekaran, "Subset selection for active object recognition," In *CVPR, , IEEE Computer Society Press*, vol. 2, pp. 511-516, June 1999.

[10] J.A.K. Suykens, and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, pp. 293-300, 1999.