SUCCESSIVE PATTERN LEARNING BASED ON TEST FEATURE CLASSIFIER AND ITS APPLICATION TO DYNAMIC RECOGNITION PROBLEMS

Yukinobu Sakata†, Shun'ichi Kaneko‡and Takayuki Tanaka‡

†JSPS Research Fellow, Graduate School of Information Science and Technology Hokkaido Univ.
‡Graduate School of Information Science and Technology Hokkaido Univ.
Kita-13 Nishi-8 Kitaku, Sapporo, Japan
sakata@ssc.ssi.ist.hokudai.ac.jp, kaneko@ssi.ist.hokudai.ac.jp, ttanaka@ssi.ist.hokudai.ac.jp
http://www.ssc-lab.com

Commission III/3

KEY WORDS: classification, test feature classifier, successive learning, dynamic recognition problem

ABSTRACT:

Fundamental characteristics of successive lerarning by Test Feature Classifier(TFC), which is non-parametric and effective with small data, are examined. In the learning, a new set of training objects, they are fed into the classifier in order to obtain a modified classifier. We propose an efficient algorithm for reconstruction of prime test features, which are combinaton feature subsets for getting the excellent performance. We apply the proposed successive TFC to dynamic recognition problems where the traning data increase successively and also characteristic of the data change with progress of time, and examine the characteristic by the experiments which used the real world data and a set of simulated data.

1 INTRODUCTION

Classification technique is one of the most important subjects in the field of pattern analysis, and many classifiers have been proposed. One of the real-world classification problems is classification of defect image from a semiconductor testing system. Semiconductor wafers are produced in production line, and some defects are included in the produced semiconductors. In the production line of semiconductor, it is important to solve sources of the defects rapidly as soon as possible. We can estimate the condition of the lines from classification of defect patterns, and repair the part, which has broken. In this system, after recognition of a set of unknown data by a classifier, they are fed into the classifier in order to obtain modified performance. As just described, in a real world situation where we need to make a high performance classifier using the added data from every day process, successive learning algorithm is necessary.

Some successive learning algorithms have been proposed for on-line handwriting recognition in order to gain better performance depending on sequential training data than the one of the classifiers in previous steps (Y.Kimura et al., 2001, K.Akiyama et al., 2001, T.Yokota et al., 2005). These learning algorithms are for adapting to a writer by learning the writer's own style significantly improve recognition accuracy. By adding the characters which the user input into the initial data set, the system is more adapted for the user. The purpose of these research is obtaining a high performance in small data and calculation by selecting addition data. It is an important point because the target of an on-line handwriting recognition system is to assimilate it into the small system which has limited resource. However, these algorithms are for a situation which we can obtain enough number of training data, these are not so effective to the semiconductor testing system. In the semiconductor testing system, we need manned inspection of high cost to gain training data, and because of inspection cost, we don't obtain enough number of training data. In such a case, we have to design a high performance classifier from limited data. Furthermore, in successive learning classification, it is often necessary to adapt to change in progress in the lines for keeping high performance.

We have proposed a novel classifier, Test feature classifier, hereafter referred to as TFC, which utilize combinatorial features(Prime Test Feature:PTF) for getting an excellent performance with some limited training data. TFC is a non-parametric classifier which has a better performance with only small set of training data and is based on the structured features. These characteristics can contribute to successive learning as follows: the non-parametricity for adaptability to data in the real world, the trainability with small data for initial construction of better classifiers in early steps, and the mutual independency of test features for easiness of local modification of the classifiers. And then we applied them to some real-world applications in order to verify their practical feasibility (V.Lashkia et al., 1999, V.Lashkia et al., 2000, V.Lashkia et al., 2001, Itqon et al., 2001, Itqon et al., 2000). In addition, we have proposed successive learning algorithm based on TFC, and applied it to a real world problem of classification of defects on wafer images, obtaining excellent performance(Y.Sakata et al., 2004, Y.Sakata et al., 2005). In TFC, extracting a set of PTFs form training data means constructing TFC. After definition of a set of PTFs, viz. training an initial TFC classifier, with prespecified data, when a set of new training data is provided, the TFC should be updated or modified in order to adjust itself to the augmented data including the new ones. It is worth noting that not all the PTFs must be extracted in case of a large dimension of features, and then a part of PTFs can be generally utilized to construct of new TFC. By using this this idea, we can save computational cost. In this paper, we modify TFC by weighted voting selection with weight coefficients, and verify the effectiveness experimentally.

2 TEST FEATURE CLASSIFIER

2.1 Outline

Since the mathematical formalization of TFC has been given in (V.Lashkia et al., 1999, V.Lashkia et al., 2000), we give a brief introduction of the classifier with qualitative and semantic explanation. Figure1 shows a basic structure of batch TFC. TFC is consisted by the learning and discrimination procedures. In the

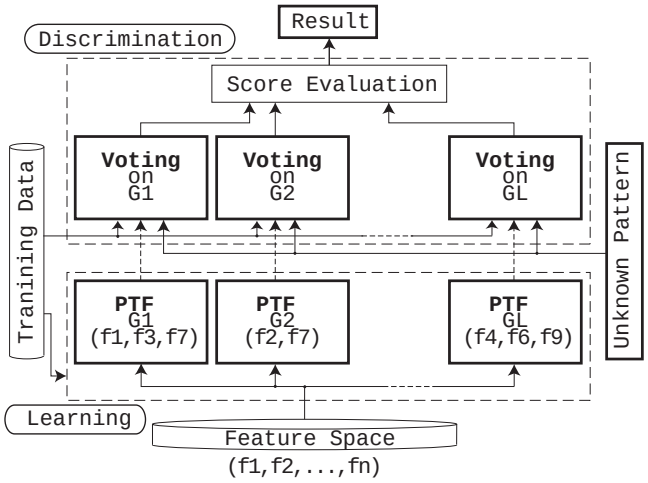

Figure 1: Outline of batch TFC.

learning procedure, the total feature space is divided into local sub-spaces of combinatorial features with overlap through a non-parametric and particular investigation, where the features joined together are called *test*, hereafter TF, or *prime test*, hereafter PTF, which is an irreducible test. In the discrimination procedure, it discriminates an input unknown pattern as candidates by checking it in each sub-space using the corresponding PTF, and then classifies the pattern by voting averaged scores in the sub-spaces. The classifier, therefore, aims to gain high performance even with a small set of training data by partial discriminations in the sub-spaces. Provided a set of training data without overlap across classes, a TF is defined as any combination of features that does not confuse every pattern only using the selected features, that is, it satisfies the condition of non-overlap. Smaller combinations, in general, might cause overlap, but are profitable to be of low cost computing. Since any combination including a TF is proved to be a TF too, we should choose irreducible TFs for discrimination, as PTFs. In Figure1, we show examples of PTF.

TFC is constructed by PTSs, which are extracted from training data.

Let $\mathcal{P} = \{G_1, G_2, \ldots, G_L\}$ be a set of L PTFs extracted, and we consider about discrimination of $\overline{t}$, here. Each class has a set of training patterns $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \cdots, \mathcal{X}_h\}\,(h \geq 2)$, and each set consists of training patterns $\mathcal{X}_i = \{{}_i\overline{x}_1, {}_i\overline{x}_2, \cdots, {}_i\overline{x}_{p_i}\}$ $(1 \leq i \leq h)$. The i-th class has p_i patterns ${}_i\overline{x}_j$ $(1 \leq j \leq p_i)$. ${}^k\overline{t}$ and ${}^k\overline{x}$ the projections of an unknown pattern and a training pattern, respectively, onto G_k. $d({}^k_i\overline{x}_j, {}^k_u\overline{x}_v)$ is the Euclidean distance between any two patterns ${}^k_i\overline{x}_j$ and ${}^k_u\overline{x}_v$. In TFC, as mentioned previously, discrimination is by voting. A voting selection is performed by calculating a score for each class by the following voting function.

$$V_i(\overline{t}) = \sum_{k=1}^{L} \sum_{j=1}^{p_i} r_i(d({}^k\overline{t}, {}^k_i\overline{x}_j)) \tag{1}$$

where, $r_i(d)$ is defined as follow

$$r_i(d) = \begin{cases} 1 & \text{if } d = \min\limits_{1 \leq j \leq p_i} d({}^k\overline{t}, {}^k_i\overline{x}_j) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

We compare unknown pattern with training patterns on G_k, and vote for the class, which a most similar training pattern belongs to. Finally, the unknown pattern $\overline{t}$ is classifyed as the class C_i, if

Table 1: Example of training data set

Class	Training patterns	Features		
		f_1	f_2	f_3
C_1	${}_1\overline{x}_1$	3	1	2
	${}_1\overline{x}_2$	0	5	1
	${}_1\overline{x}_3$	2	3	5
	${}_1\overline{x}_4$	3	2	0
C_2	${}_2\overline{x}_1$	2	4	3
	${}_2\overline{x}_2$	4	5	4
	${}_2\overline{x}_3$	4	5	3

Table 2: Example of TF and PTF

Feature Combination	TF	PTF
$G_1 = \{f_1\}$	×	×
$G_2 = \{f_2\}$	×	×
$G_3 = \{f_3\}$	○	○
$G_4 = \{f_1, f_2\}$	○	○
$G_5 = \{f_1, f_3\}$	○	×
$G_6 = \{f_2, f_3\}$	○	×

$V_i(\overline{t})$ fills following condition.

$$V_i(\overline{t}) > V_u(\overline{t}), \quad \forall u(\neq i) \tag{3}$$

2.2 Numerical example

Table1 shows a set of training data of three dimensional features in two classes. In this simple case, all the combinations of features, $6 = 2^3 - 2$ combinatorial features except for the null and complete combinations, can be checked if each of them is a TF. TF first $\{f_1\}$ raises confusion because ${}_1\overline{x}_3$ in $\mathcal{C}_1$ and ${}_2\overline{x}_1$ in $\mathcal{C}_2$ have the same reduced or projected representation on $\{f_1\}$, hence $\{f_1\}$ is not a TF. On the other hand, we compare $\mathcal{C}_1$ with $\mathcal{C}_2$ on $\{f_3\}$, there is no representation that overlap between two classes, hence $\{f_3\}$ is a TF. We find,otherwise,$\{f_1, f_2\}$, $\{f_1, f_3\}$,etc., are TFs. Next, we check if each of TFs is a PTF. For example, G_4 is a PTF because all of the combinatorial features made by G_4 except G_4 itself are not TFs. On the other hand, G_5 is not a PTF because it contains another TF G_3. Table2 shows two PTFs G_3 and G_4 in total. Table3 shows an example of discrimination of an unknown patterns $\overline{t} = (4, 5, 1)$by voting through the PTFs G_3 and G_4, resulting the pattern classified into C_2.

2.3 Successive TFC

After definition of a set of PTFs, viz. training an initial TFC classifier, with prespecified data, when a set of new training data is provided, the TFC should be updated or modified in order to adjust itself to the augmented data including the new ones. It is worth noting that not all the PTFs must be extracted in case of a large dimension of features, and then a part of PTFs can be generally utilized to construct TFCs of enough performance.

We propose a novel algorithm for modifying the set of PTFs depending on the newly provided training data below, where a partial and efficient computation is possible. Let $\mathcal{C}_n$ be the classifier after n times learning, $\mathcal{X}_n$ the set of accumulated training data, $\mathcal{T}_n$ the set of TFs, $\mathcal{P}_n$ the set of PTFs, and $\mathcal{N}_n$ the set of non-test combination of features, where $\mathcal{C}_n = (\mathcal{X}_n, \mathcal{P}_n)$ and $\mathcal{F} = \mathcal{T}_n + \mathcal{N}_n$, $\mathcal{P}_n \subseteq \mathcal{T}_n$ with the total set of features $\mathcal{F}$. We define an indicating function $\phi_{\mathcal{P}}(G, \mathcal{X})$ to judge if any combination

Table 3: Discrimination of unknown pattern $\overline{t} = (4, 5, 1)$ by voting

		PTF		V
		$G_3(f_3)$	$G_4(f_1, f_2)$	
C_1	${}_1\overline{x}_1 = [3, 1, 2]$	-	-	1
	${}_1\overline{x}_2 = [0, 5, 1]$	1	-	
	${}_1\overline{x}_3 = [2, 3, 5]$	-	-	
	${}_1\overline{x}_4 = [3, 2, 0]$	-	-	
C_2	${}_2\overline{x}_1 = [2, 4, 3]$	-	-	2
	${}_2\overline{x}_2 = [4, 5, 4]$	-	1	
	${}_2\overline{x}_3 = [4, 5, 3]$	-	1	

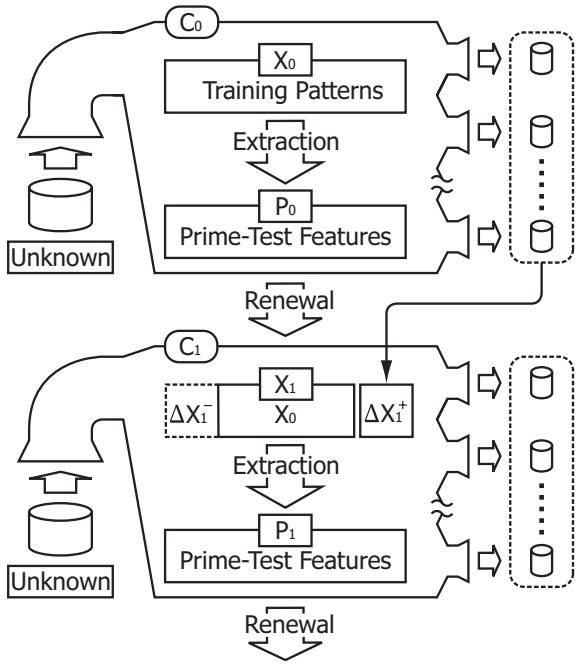

Figure 2: Successive modification of training patterns and PTF.

of features G is PTF or not.

$$\phi_{\mathcal{P}}(G, \mathcal{X}) = \begin{cases} 1 & \text{if } G \in \mathcal{P} \\ 0 & \text{if } G \notin \mathcal{P} \end{cases} \quad (4)$$

This function is a convention to get clear description of the algorithm, and it can be realized by a program for investigating all the training data on the inputs G and $\mathcal{X}$. We define a successive algorithm for learning. Figure2 shows the outline of the algorithm.

Successive learning algorithm

1. **Initialization** n=0. Provide $\mathcal{X}_0$, then extract $\mathcal{P}_0$. Construct an initial classifier $\mathcal{C}_0 = (\mathcal{X}_0, \mathcal{P}_0)$.

2. **Classification** Discriminate unknown data $\mathcal{U}_n$ by $\mathcal{C}_n$. After inspecting the result, determine the additional data $\Delta\mathcal{X}^+_{n+1} \subseteq \mathcal{U}_n$. If there are data that should be deleted, determine the deletional data $\Delta\mathcal{X}^-_{n+1} \subseteq \mathcal{X}_n$.

3. **Modification** Construct $\mathcal{X}_{n+1}$=$\mathcal{X}_n + \Delta\mathcal{X}^+_{n+1} - \Delta\mathcal{X}^-_{n+1}$.

4. **Elimination of PTF** Through PTF check, construct ${\mathcal{P}'}_n = \mathcal{P}_n - \{G \mid \phi_{\mathcal{P}}(G, \mathcal{X}_{n+1}) = 0\}$ as available PTFs in the next step.

5. **Incremental definition of PTF** Construct $\Delta\mathcal{P}_{n+1}$ as a new set of PTFs, and then introduce it into $\mathcal{P}_{n+1}$ as ${\mathcal{P}'}_n + \Delta\mathcal{P}_{n+1}$. (A detailed algorithm is described in the next section.)

6. **Termination** Check the performance of $\mathcal{C}_{n+1} = (\mathcal{X}_{n+1}, \mathcal{P}_{n+1})$. If enough, terminate. Otherwise, go back to Step 2 after update $n \leftarrow n + 1$.

2.4 Incremental definition of PTF

In Step 5 of the basic algorithm abovementioned, $\Delta\mathcal{P}_{n+1}$, a new set of PTFs, has to be constructed, therefore, we design an efficient algorithm for new PTFs adapted to updated training data. The efficiency comes from inspecting not all the combinatorial features.

In the case of renewal of PTF by the addition of data, for any G, it is enough to find those that satisfy $\phi_{\mathcal{P}}(G, \mathcal{X}_{n+1}) = 1$. We know, otherwise, any G that is not a TF on a partial set of training data is not a TF on the updated set including the partial set. Hence, if G is not a TF on $\mathcal{X}_n$, it is not a TF on $\mathcal{X}_{n+1}$, viz. theoretically, if $\mathcal{X}_n \subseteq \mathcal{X}_{n+1}$ and $\phi_{\mathcal{P}}(G, \mathcal{X}_n) = 0$, we know $\phi_{\mathcal{P}}(G, \mathcal{X}_{n+1}) = 0$ holds. Therefore, we don't have to extract new PTF from all of ordinary combinations in $\mathcal{N}_n$, hence we can limit our focus the inspection only within $\mathcal{T}_n$ for new PTFs.

In the case of renewal of PTF by the deletion of data, a basic idea is similar to the update by the addition of data, we can calculate efficiently by limit the focus of the search of PTF. In this case, contrary to the case of the addition , candidates of new PTF appear in $\mathcal{N}_n$. Hence we can limit our focus on the inspection only within $\mathcal{N}_n$ for new PTFs.

2.5 Verification experiment

Verification experiment has been performed in order to confirm effectiveness of the proposed algorithm. We compare STFC with TFC about the calculation time. We utilized the four set of "Y", "U", "K", and "I" from the well known character feature set Letter(P.M.Murohy and D.W.Aha, 1994) as $p = 16$ dimensional training data, so the class number is $h = 4$. In updating by addition of data, an initial classifier $\mathcal{C}_0$ was trained by 60 data, four sets of 15 data for each character. Through 50 steps of successive learning, 40 data, four sets of 10 data, have been incrementally added each time. In updating by deletion of data, an initial classifier $\mathcal{C}_0$ was trained by 2060 data, four sets of 515 data for each character. Through 50 steps of successive learning, 40 data, four sets of 10 data, have been decreasingly added each time. We experimented about the case of $p = 16$, $p = 13$, and $p = 10$. Figure 3,and Figure4 show the results. We found the calculation time of STFC is shorter than that of TFC in all learning stages. By using the successive TFC,we can save about 70% on the total computational cost in comparison with a batch learning.

3 WEIGHTED PTF

3.1 Weighted voting selection

As shown in Equations(1) and (2), in the discrimination procedure of TFC, each PTF votes an equal score. Though, the distribution of training patterns on each PTF, subset of the full feature space, may be different due to bias or non-uniform dispersion in

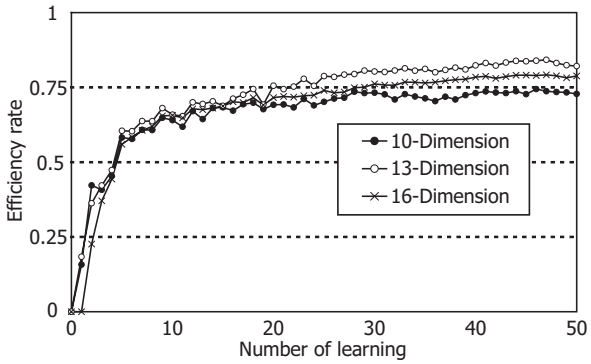

Figure 3: Comparison about calculation time (data addition)

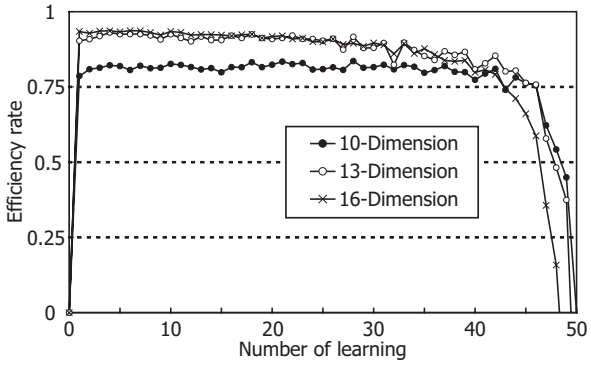

Figure 4: Comparison about calculation time (data deletion)

distribution, resulting difference in discrimination performances of PTF's. In order to utilize these particularities of PTFs more effectively in descrimination, in Equation (1), we propose a scheme for weighted voting selection by weight coefficients as follows:

$$V_i(\overline{t}) = \sum_{k=1}^{L} \sum_{j=1}^{p_i} {}^k w_i r_i(d({}^k\overline{t}, {}_i^k\overline{x}_j)) \tag{5}$$

Each PTF G_k has a weight coefficient ${}^k w_i$ with respect to each class C_i, which is defined as

$${}^k w_i = \frac{\sum_{j=1}^{p_i} {}^k\lambda_i(j)}{p_i} \tag{6}$$

where, ${}^k\lambda_i(j)$ is a vertual indicating function which determines whether any training pattern ${}_i\overline{x}_j$ belonging to the class C_i has a nearest neighbor training pattern except for itself which is a member of C_i too, or not. We can formalize the above indicating condition as

$${}^k\lambda_i(j) = \begin{cases} 1 & \text{if } {}_i^k ds_j < {}_i^k dd_j \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where these distances ${}_i^k ds_j$ and ${}_i^k dd_j$ are defined as

$${}_i^k ds_j = \min_{1 \le v \le p_i} d({}_i^k\overline{x}_j, {}_i^k\overline{x}_v), \quad v \neq j \tag{8}$$

$${}_i^k dd_j = \min_{1 \le u \le h, 1 \le v \le p_u} d({}_i^k\overline{x}_j, {}_u^k\overline{x}_v), \quad u \neq i \tag{9}$$

In the discrimination procedure of TFC, an unknowm pattern is classified in terms of each PTF by the nearest neighbour rule, and finaly, the class which obtains the highest support from PTFs becomes the final classification. The weight coefficient ${}^k w_i$ is calculated as a recognition rate with respect to the class C_i through testing by the leave-one-out method with nearest neighbours on the subset G_k, and it means the reliability of PTF G_k when it supports the class C_i. Figure5 shows an example of the distribution of training patterns on a PTF G_k. In Figure5(a), there are no overlapping across classes, while in Figure5(b), there is severe overlap between Class 1 and Class 3. Weight coefficients of the case of Figure5(a) were $({}^k w_1 = 0.96, {}^k w_2 = 1.00, {}^k w_3 = 0.99)$, and the ones in the case of Figure5(b) were $({}^k w_1 = 0.68, {}^k w_2 = 0.99, {}^k w_3 = 0.63)$. Generally, ${}^k w_i$ has lower values for a class which overlaps with other classes, and higher values for classes which have less overlap with other classes. Hence, the PTF G_k votes a high score for the class with a high possibility that the PTF can correctly discriminate the true classes, and the PTF votes a lower score when it votes to the class with a high possibility that it can discriminate correctly. By using the weight coefficients in weighted votes, we can construct another type of the successive TFC, in which each elemental classifier correponding to each PTF reflects the relative and substancial capacity in descrimination. In dynamic recognition problems, the distribution of training patterns in the feature space changes with progress of time. In this situation, some PTFs which have been effective for discrimination in early stages in learning become not effective in later learning stages. Voting from such kind of PTFs causes performance degradation of the successive TFC. However, by using the weight coefficients abovementioned, we can provide some limit in voting through the low performance PTFs.

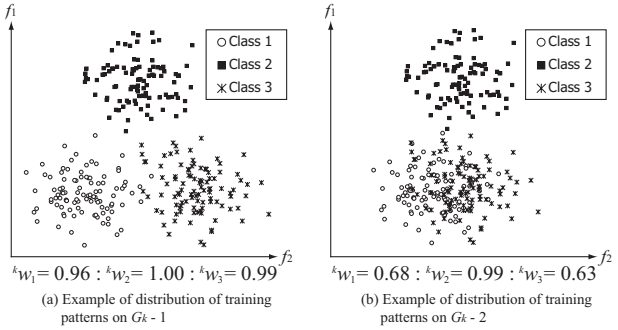

Figure 5: Difference of weight coefficient by pattern distribution

3.2 Evaluation experiments by defect images

Evaluation experiments have been performed with defect images in order to confirm the effectiveness of the weighted PTF. In production lines of semiconductor, it has been important to inspect microscopic defects occurred on wafers by a scanning electronic magnifier, SEM, in order to infer the sources of the defects as rapidly as possible. Classification of defect patterns recently becomes necessary for estimating conditions of the production lines and restoring the perfomence in vibo.

Fig.6 shows representative examples of defect images on semiconductor wafers in nine classes, which were taken by a SEM. The problem here is to classify them into the classes by use of 662 training data. Each pattern consists of ten features including popular ones, such as shape, texture in pattern regions and backgrounds, and difference between such quantities. The range of these features was [0,1], and they had an fixed point expression with the six effective digits.

TFC was trained by 165 training data through random selection from 662 data. The number of training data in each class was set as constant, and it has not changed in each trial. Performance evaluation was made by use of the rest of 497 data. Performance with the weight coefficients was compaired with performance of

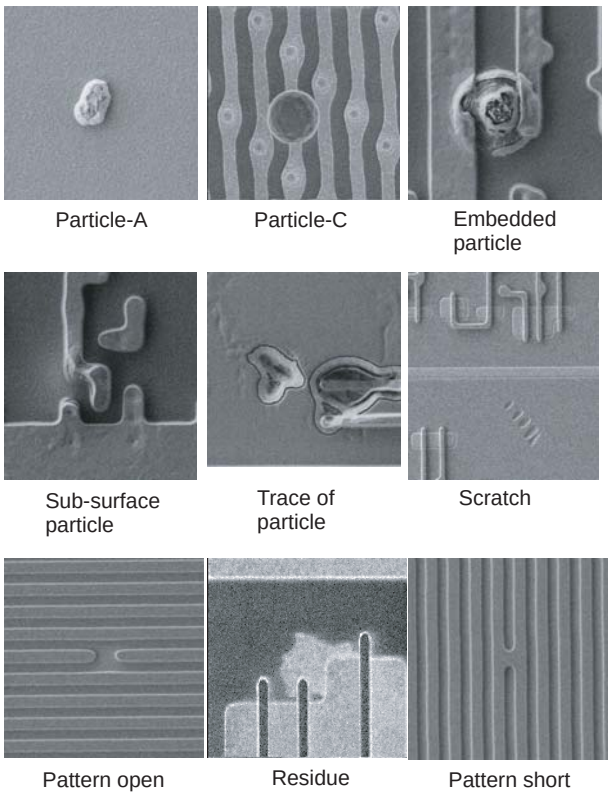

Figure 6: Examples of SEM defect images (magnified subimages).

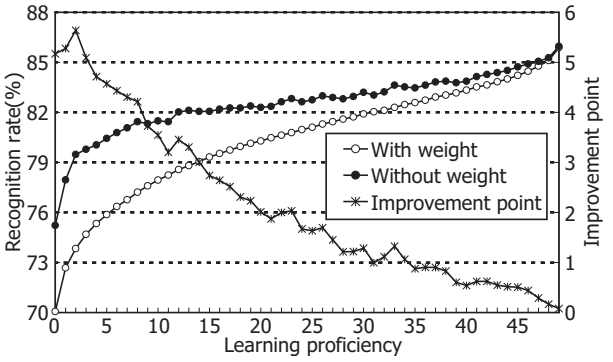

Figure 7: Learning proficiency and improvement point.

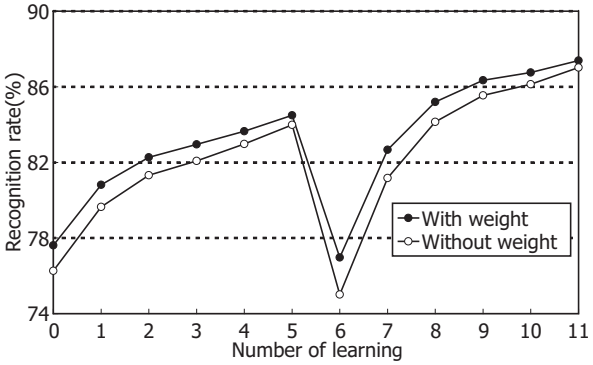

Figure 8: Comparison with other classifiers.

without the weight coefficient in 5000 trials. We sorted the results with respect to recognition rates by the classification without the weight coefficient into ascending order. Figure7 shows the results of averaging over every 100 results. We can clearly find improvement in performance by introducing the weight coefficients, and then we can also find the weighted voting selection is effective especially when the learning proficiency is low. This will be one of the favorable characteristics, for example, in case of unskilled classifiers or an early learning stage, where we can't get enough number of training data. Even in this case, the results shown in Figure7 will help you to obtain revised versions of classifiers not by increasing training data but by sophisticating each PTF with the weight coefficient. Furthermore, it is rather notable for the improvement in recognition rate to have the larger value as we have the lower learning proficiency. It shows that the proposed classifier is more effecitve in the cases of immature classifiers, and therefore we may have large merit especially in the real industries with the proposed weighted classifiers.

4 DYNAMIC RECOGNITION PROBLEM

4.1 Problem definition

We call in this study a recognition problem which the characteristic of data changes with progress of time as Dynamic recognition problem. In classification of the defect images of semiconductor wafers, which is one of the real-world classification problems, the characteristics of the defects change because of somewhat physical changes of the production line. In addition, the characteristics of the data change through updating classification categories, integration, addition, deletion, and modification in classification strategy. In these situations, we have to always maintain high performance, and need the successive learning algorithm proposed by this research is necessary.

For this problem, we have proposed in the previous sections the learning method possible to adapt to the problem which the characteristic of data changes with progress of time.

4.2 Experiments

We apply the proposed successive TFC to dynamic recognition problems. Experiments have been performed with defect images. These are the same data as being used in 3.2. It is known that in the production lines of semiconductors, we must have dynamic recognition problems, although it is understood that such kind of data are not so easy to sample from the real production lines. Therefore, in this experiment, we use a simulated data obtained as follows: we devided all the data into two data sets, D-1 and D-2, from which the characteristics of distributions are different each other. In order to simulate dynamic recognition problems, we changed additional data with progress of time in the order D-1 $\rightarrow$ D-2.

Figure8 shows the result of comparning the cases of "with weight coefficients" to "without weight coefficients". The averaged performance in each method was drawn from 100 trials with different combinations of the training data. An initial classfier $\mathcal{C}_0$ was trained by 41 data from D-1, and then. addition was done with 41 data from D-1 at each step till $n = 5$. In the steps from $n = 6$ to $n = 11$, the added data were changed from D-1 to D-2. The performance in $n = 0$ to $n = 5$ was evaluated always with the constant 80 data from D-1, in $n = 6$ to $n = 11$ was evaluated always with the constant 80 data from D-2. Both methods had a similar tendency of growth in recognition rates, and the performance of "with weight coefficients" was better than that of "without weight coefficients".

5 CONCLUSIONS AND FUTURE WORK

A novel algorithm has been proposed for successive learning based on the non-parametric pattern classifier Test feature classifier TFC. Using the successive learning algorithm, we can save about 70% on the total computational cost in comparison with a batch learning. The performance of TFC was modified by weighted voting selection with weight coefficient. It is effective especially in early learning stage. Successive TFC has been applied to dynamic recognition problems.

As future tasks to be considered, strategies for deletion of training data should be proposed and applied to real-world problems.

REFERENCES

Itqon, S.Kaneko, S.Igarashi and V.Lashikia, 2000. Extended test feature classifier for many-valued patterns and its experimental evaluations. Trans. IEE of Japan 120-C(11), pp. 1762–1769.

Itqon, S.Kaneko, S.Igarashi and V.Lashkia, 2001. Multi-class test feature classifier for texture classification. Malaysian Journal of Computer Science 14(1), pp. 83–953.

K.Akiyama, N.Iwayama, H.Tanaka and K.Ishigaki, 2001. An adaption method based on template cache for online character recognition. PRMU2000 105(701), pp. 69–76.

P.M.Murohy and D.W.Aha, 1994. Uci repository of machine learning database. University of California-Irvine.

T.Yokota, S.Kuzunuki, N.Hamada, K.Katsura and M.Nakagawa, 2005. An on-line handwritten character recognition system with candidate-selection triggered learning by the add and average method. IEICE, D-II J88-DII(3), pp. 552–561.

V.Lashkia, S.Kaneko and M.Okura, 2001. On high generalization ability of test feature classifiers. Trans. IEEJ 121-C(8), pp. 1347–1353.

V.Lashkia, S.Kaneko and S.Aleshin, 1999. Textual region location in complex images using test feature classifiers. Canadian Journal Electronic & Computer Engineering.

V.Lashkia, S.Kaneko and S.Aleshin, 2000. Distance-based test feature classifiers and its applications. IEICE Trans. Inf. & Syst. E83-D(4), pp. 904–913.

Y.Kimura, K.Okuda, A.Suzuki and M.Sano, 2001. Analysis and evaluation of dictionary learning on handy type pen-input interface for personal use. IEICE, D-II J84-DII(3), pp. 509–518.

Y.Sakata, S.Kaneko, Y.Takagi and H.Okuda, 2004. Successive pattern learning based on test feature classifier and its application to defect image classification. IEEJ Trans.EIS 124(3), pp. 689–698.

Y.Sakata, S.Kaneko, Y.Takagi and H.Okuda, 2005. Successive pattern classification based on test feature classifier and its application to defect image classification. Pattern Recognition 38(11), pp. 1847–1856.