Novel Approximate Statistical Algorithm for Large Complex Datasets

Yohei Takeuchi, Momoyo Ito, and Minoru Fukumi

***Abstract*—In the field of pattern recognition, principal component analysis (PCA) is one of the most well-known feature extraction methods for reducing the dimensionality of high-dimensional datasets. Simple-PCA (SPCA), which is a faster version of PCA, performs effectively with iterative operated learning. However, SPCA might not be efficient when input data are distributed in a complex manner because it learns without using the class information in the dataset. Thus, SPCA cannot be said to be optimal from the perspective of feature extraction for classification. In this study, we propose a new learning algorithm that uses the class information in the dataset. Eigenvectors spanning the eigenspace of the dataset are produced by calculating the data variations within each class. We present our proposed algorithm and discuss the results of our experiments that used UCI datasets to compare SPCA and our proposed algorithm.**

***Index Terms*— Pattern recognition, principal component analysis, supervised learning.**

I. Introduction

Eigenspace models play an important role as feature extraction methods in pattern recognition [1]. Principal Component Analysis (PCA) [1] is one of the most well-known feature extraction methods and is effective in reducing the dimensionality of input datasets. PCA computes its eigenspace using the eigenvalue decomposition of a covariance matrix from the datasets. All data are used simultaneously in order to encourage the eigenvalue decomposition to produce eigenvectors that spans the eigenspace. However, solving the eigenvalue problem for input data with a large number of dimensions is computationally expensive.

Therefore, various approaches to reduce the computational cost of PCA's learning algorithm have been proposed. Simple-PCA (SPCA) [3], which was proposed by Partridge et al., is a faster version of PCA that does not calculate the covariance matrix. SPCA requires only iterative computations to produce the eigenvectors. It uses an approximation algorithm in which principal components are sequentially identified, beginning with the first component. SPCA has been proved effective in many applications including handwritten character recognition, dimensionality reduction for information retrieval models, and facial image recognition [4-11]. The algorithm used in SPCA sequentially solves for eigenvectors that maximize the variance over all samples. However, SPCA is not particularly effective in cases where the input datasets are distributed in a complex manner.

In this study, we propose a new learning approach, Class-included SPCA (CSPCA), which uses the class information in the input dataset. For classification, the eigenspace spanned by the eigenvectors must have a high degree of separability. CSPCA produces eigenvectors that maximize the variance of the input data in each class. The idea behind CSPCA is that as it learns, the eigenvector maximizes the variance when an input datum belongs to the same class, which belongs to the eigenvector, and it reduces the projected value in the eigenvector if the datum belongs to a different class. Thus, CSPCA learns features that separate the data belonging to each class, and the number of the eigenvectors that are learned by CSPCA is the same as of the number of classes in the input dataset. In addition, CSPCA's computational cost for leaning eigenvectors is equivalent to the cost of SPCA. The CSPCA algorithm is based on the SPCA algorithm with a simple modification that avoids algorithmic complexity.

The rest of this paper is organized as follows: Section 2 describes the SPCA algorithm. Section 3 presents the details of our CSPCA algorithm, which uses class information about the input dataset to learn the eigenspace. Section 4 presents the results of computer simulations that use datasets from the UCI Machine Learning Repository [12]. Finally, Section 5 presents our conclusions and future research directions.

II. Simple-PCA (SPCA)

SPCA [3] is an iterative statistical approach that was proposed by Partridge et al. to quickly learn the eigenspace spanned by eigenvectors. SPCA uses an approximation algorithm in which principal components are learned in a sequential order, beginning with the first component. SPCA has been proved effective in many applications including handwritten character recognition, and facial image recognition. The algorithm used in SPCA sequentially solves for eigenvectors that maximize the variance over all samples. The details of the algorithm are as follows:

First, the set of input vectors is defined as

$$\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N] \quad (1)$$

and all input data are centered to produce

$$\boldsymbol{X}' = [\boldsymbol{x}'_1, \cdots, \boldsymbol{x}'_N] \quad (2)$$

The manuscript received on July 17, 2012; revised September 3, 2012.

Y. Takeuchi is a doctoral course student with the Graduate School of Advanced Technology and Science, the University of Tokushima, Tokushima, 770-8506 Japan (e-mail: takeuchi-yohei@is.tokushima-u.ac.jp).

M. Ito is an Assistant Professor with the Department of Information Science and Intelligent Systems, the University of Tokushima, Tokushima, 770-8506 Japan (e-mail: momoito@is.tokushima-u.ac.jp).

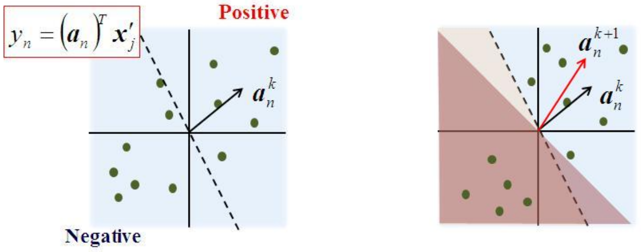
M. Fukumi is a Professor with the Department of Information Science and Intelligent Systems, the University of Tokushima, Tokushima, 770-8506 Japan (e-mail: fukumi@is.tokushima-u.ac.jp).

10.7763/IJMLC.2012.V2.222

where *N* is the number of input data. The output function is defined as

$$y_n = \left(\boldsymbol{a}_n^k\right)^T \boldsymbol{x}'_j \tag{3}$$

where $\boldsymbol{a}_n^k$ is an eigenvector that represents the *n*th principal component and *k* is the number of repetitions that have been completed. The initial vector $\boldsymbol{a}_n^0$ is set to be a random vector. If the input vector component $\boldsymbol{x}'_j$ has the same direction as $\boldsymbol{a}_n^k$, then the output function (3) outputs a positive value, and if it has the opposite direction, then the output function outputs a negative value [Fig. 1(a)].



(a) Output function (b) Eigenvector replacement

Fig. 1. Illustrations of the SPCA algorithm

The following threshold function is introduced for this purpose:

$$f\left(y_n, \boldsymbol{x}'_j\right) = \begin{cases} \boldsymbol{x}'_j & if\ y_n \geq 0 \\ -\boldsymbol{x}'_j & otherwise \end{cases} \tag{4}$$

The initial random vector $\boldsymbol{a}_n^0$ brought closer to the same direction that the vector $\boldsymbol{a}_n$ possesses through the use of these functions and repetition of the following operation.

$$\boldsymbol{a}_n^{k+1} = \frac{\sum_j f\left(y_n, \boldsymbol{x}'_j\right)}{\left\|\sum_j f\left(y_n, \boldsymbol{x}'_j\right)\right\|} \tag{5}$$

where $\boldsymbol{a}_n^{k+1}$ is the eigenvector after *k*+1 repeated calculations. The value of the output function is calculated by using $\boldsymbol{a}_n^k$, which is the previous calculation result. These repeated calculations will continue until $\boldsymbol{a}_n^{k+1}$ converges [Fig. 1(b)].

Before calculating the next eigenvector, a new vector $\hat{\boldsymbol{x}}_j$ has to be calculated so that the eigenvectors satisfy the orthogonality condition. The vector is calculated as follows:

$$\hat{\boldsymbol{x}}_j = \boldsymbol{x}'_j - \left(\boldsymbol{a}_n^{k+1} \cdot \boldsymbol{x}'_j\right)\boldsymbol{a}_n^{k+1} \tag{6}$$

After the component is removed, the principal component can be evaluated by repeating the same calculation in order to produce a high accumulated relevance.

III. Class-Included SPCA (CSPCA)

SPCA can calculate eigenvectors faster than the original matrix-based PCA. However, although both methods maximize the variance in the eigenspace, neither can achieve the recognition accuracy that is necessary for classification. Therefore, they might not be efficient from the perspective of feature extraction for classification.

We propose Class-included SPCA (CSPCA) as a method that not only maximizes variance, but also improves the projected features in the eigenspace for classification. The idea behind CSPCA is that it learns an eigenvector for a specified class. Thus, the number of the eigenvectors learned by CSPCA is the same as the number of classes in the input data.

Moreover, CSPCA can learn eigenvectors with computation costs that are equivalent to the computation cost of the SPCAs. The CSPCA algorithm is based on the SPCA algorithm with a simple modification that avoids algorithmic complexity. The algorithm is described below:

First, the input dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m\}$ is obtained. Second, centering calculations areperformed in the SPCA algorithm, Third, *n*th eigenvector $\boldsymbol{a}_n^k$ $(n = 1, \cdots, C)$, where *C* is the number of classes, is randomly set, is randomly set. Finally, the output function is defined as follows:

$$f\left(y_n, \boldsymbol{x}'_j\right) = \begin{cases} \left(\left(\boldsymbol{a}_n^k\right)^T \cdot \boldsymbol{x}'_j\right)\boldsymbol{x}'_j \\ \qquad if\ \boldsymbol{x}'_j \in \omega_n \\ 0 \qquad otherwise \end{cases} \tag{7}$$

where ω_n denotes class *n*.

The abovementioned equation indicates that if the input datum, $\boldsymbol{x}'_j$, belongs to class *n*, then the *n*th eigenvector spins toward $\boldsymbol{x}'_j$. On the contrary, if the datum belongs to a different class, then learning remains uninfluenced. This is the key point of the CSPCA algorithm. By this calculation, each eigenvector is learned in a way that maximizes the variance of each class. For higher recognition accuracy for classification, a second output function is defined as follows:

$$f\left(y_n, \boldsymbol{x}'_j\right) = \begin{cases} \left(\left(\boldsymbol{a}_n^k\right)^T \cdot \boldsymbol{x}'_j\right)\boldsymbol{x}'_j & if\ \boldsymbol{x}'_j \in \omega_n \\ \left(\boldsymbol{x}'_j - \left(\left(\boldsymbol{a}_n^k\right)^T \cdot \boldsymbol{x}'_j\right)\boldsymbol{a}_n^k\right) & otherwise \end{cases} \tag{8}$$

This equation uses linear discriminant analysis to make each of the eigenvectors effective in extracting the features of each class (Fig. 2).

The upper calculation in equation (8) ensures that the eigenvector maximizes the variance of the distribution of data that are labeled with the class that it is learning. In contrast, the lower calculation in equation (8) ensures that the eigenvector is further separated from input data that belongs to other classes. In this case, the vertical component is extracted from both input datum and eigenvector and is subtracted from input. Using this procedure, CSPCA learns the eigenvector for a specified class of data.

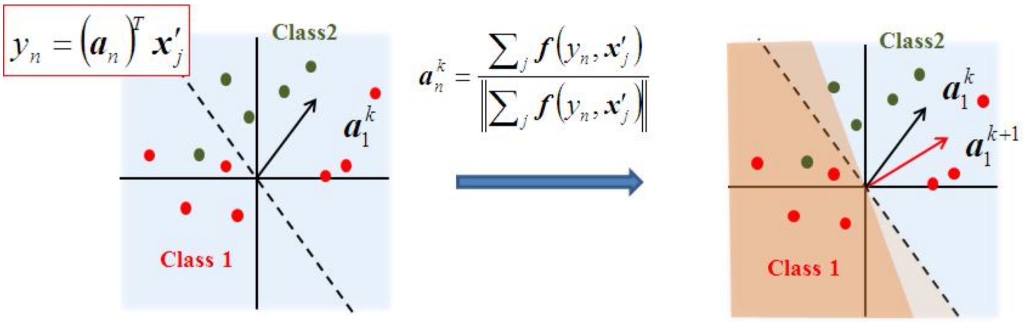

$y_n = (a_n)^T x'_j$
Class2
a_1^k
Class 1
$a_n^k = \frac{\sum_j f(y_n, x'_j)}{\left\| \sum_j f(y_n, x'_j) \right\|}$
Class2
a_1^k
a_1^{k+1}
Class 1

Fig. 2. CSPCA algorithm illustrations (2-class problem)

After calculating the output function (8), the eigenvector is reset by equation (5), as it is in SPCA. The next eigenvector is then calculated in the increasing order.

These are the procedures for learning the eigenvectors spanning the eigenspace. As previously mentioned, the number of eigenvectors is the same as the number of classes in the input dataset. This means that each eigenvector is learned for the data that are labeled as the same class.

Finally, the computational cost of CSPCA is *O*(*CN*), while the cost of SPCA is *O*(*dN*), where *d* is the number of the eigenvectors spanning the eigenspace. Thus, CSPCA can be performed as efficiently as SPCA without complex calculations.

IV. Experimental Results

In this section, we present our experimental results. We used seven datasets from the UCI Machine Learning Repository [12]. Table 1 shows the details of each dataset, including the number of dimensions, the number of classes, and the size of the dataset. The 5-fold cross-validation method was used to evaluate each dataset. We divide the Segmentation and Isolet datasets into a training dataset and a test dataset. In addition, we select the first *d* principal components for SPCA learning, and finally, we define the accumulation ratio as follows:

$$A_C(d) = \frac{\sum_{i=1}^{d} \lambda_i}{\sum_{i=1}^{m} \lambda_i} \tag{9}$$

where λ_i denotes the *i*th eigenvalue in the eigenspace. The accumulation ratio $A_C(d)$ shows how much information will be left in the eigenspace compared with the information in the input space. Furthermore, we observe that the number of dimensions of the *d* eigenvalues is larger than a certain predetermined threshold θ. In this experiment, we set $\theta = [0.8, 0.85, 0.9, 0.95, 0.99]$. It shows the best result by selecting the optimal parameters among these thresholds.

We used the Nearest Neighbor (NN) as a classifier for comparing the performances of the SPCA and CSPCA algorithms. This allowed us to rigorously compare the performances of the feature extraction methods. Using the UCI datasets for our recognition experiments, we performed the experiments 10 times with randomly sorted training data calculated the average results. The recognition accuracy and the corresponding number of eigenvectors obtained for each dataset are shown in Fig. 3 and 4, respectively.

TABLE I: Details of UCI Datasets

Name	#. of dim.	#. of classes	#. of data (train/test)
Iris	4	3	150
Wine	13	3	178
Segmentation	19	7	210/2100
Spambase	57	2	4601
Arrhythmia	279	16	452
Isolet	617	26	6238/1559
Advertisement	1558	2	3279

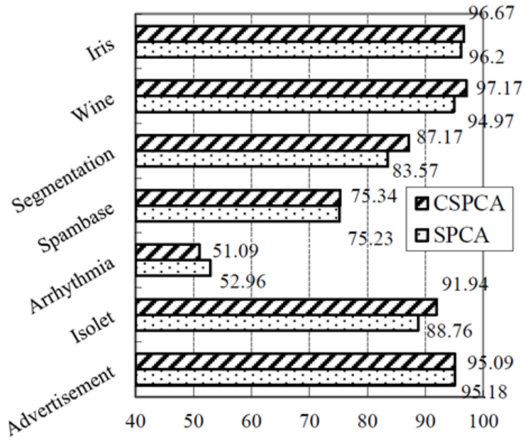


Fig. 3. Recognition Accuracy

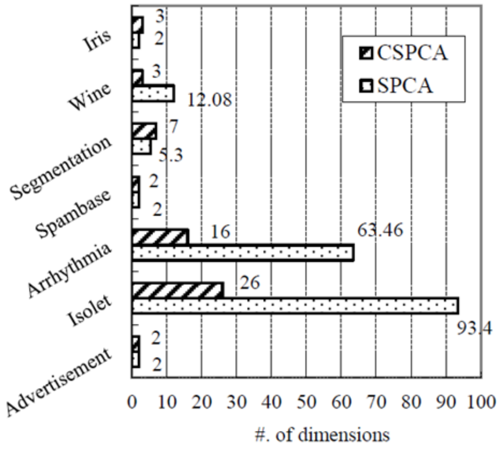


Fig. 4. Number of eigenspace dimensions

As observed in the above figures, the CSPCA features are as effective as the SPCA features. For the Wine and Isolet datasets, CSPCA's recognition accuracy is higher, despite a much smaller number of eigenvectors. This means that the distribution of features in the eigenspace is quite superior for classification. In contrast, SPCA cannot extract effective features and its memory cost is greater. Figure 5 shows the recognition accuracy and the number of dimensions for all thresholds θ using the Wine dataset.

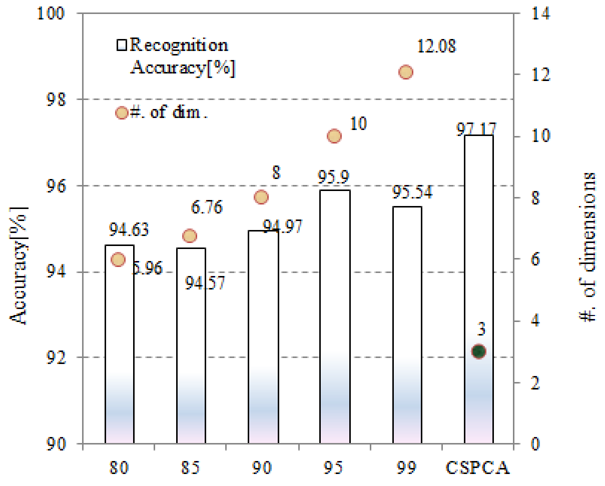


Fig. 5. Recognition accuracy and number of dimensions for SPCA and CSPCA using the Wine dataset

It is clear from Fig. 5 that the number of eigenvectors used by CSPCA is much smaller than the number used by SPCA. This result shows the effectiveness of our proposed CSPCA algorithm.

However, CSPCA does not always extract effective features for recognition. Figure 6 shows our experimental result for the Segmentation dataset, which was originally divided into a training dataset and a test dataset.

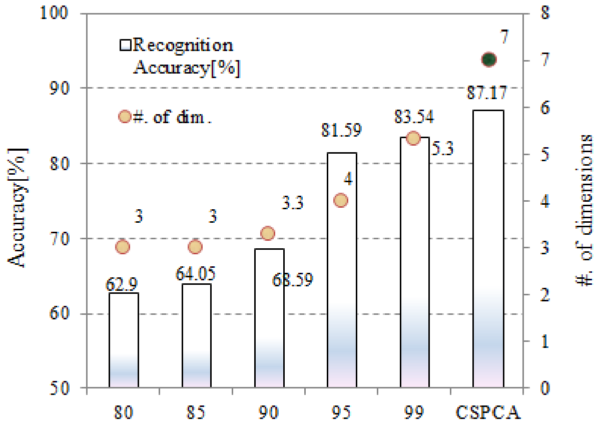


Fig. 6. Recognition accuracy and number of dimensions with each threshold θ for SPCA and CSPCA by using the Segmentation dataset

As shown in Figure 6, the features learned by CSPCA in this case were most effective for classification but with a greater number of dimensions than any of the other cases. Although the recognition accuracy of CSPCA in this case is higher than that of SPCA, CSPCA did not efficiently extract features from the Segmentation dataset.

Fig. 7 shows the recognition accuracy and the number of dimension with each threshold θ for SPCA and CSPCA using the Isolet dataset.

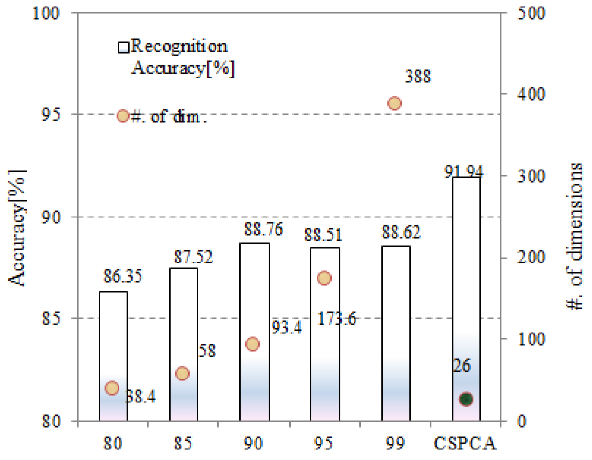


Fig. 7. Recognition accuracy andnumber of dimensions with each threshold θ for SPCA and CSPCA using the Isolet dataset

Fig. 7 shows that recognition accuracy and the number of eigenvectors of CSPCA were very good for the Isolet dataset. The algorithm used in CSPA extracted efficient features for this dataset

Finally, Fig. 8 shows the results for the Advertisement dataset.

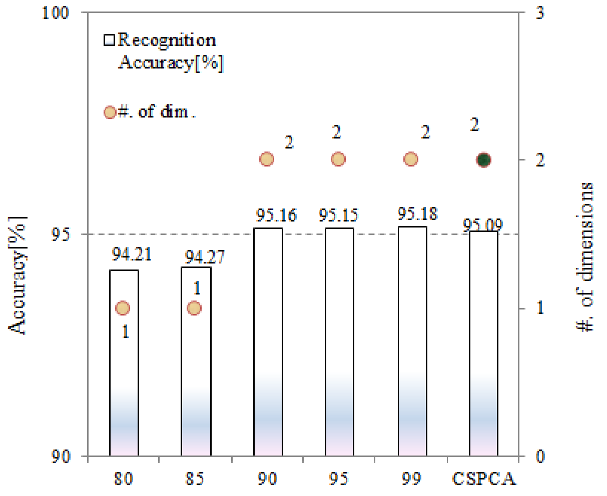


Fig. 8. Recognition accuracy and number of dimensions with each threshold θ for SPCA and CSPCA using the Advertisement dataset

As observed in Figure 8 the performance of CSPA is similar to that of SPCA. In this particular dataset, there is much variance in a low-dimensional space despite of high input dimensionality. However, this result was expected and shows the characteristics of our proposed CSPCA algorithm. In fact, CSPCA was proposed for datasets whose input space is multicolinear and can serve as a better feature extraction method in cases where SPCA cannot extract efficient features for classification.

V. Conclusions

This paper presented our proposed CSPCA algorithm, which is a modified SPCA algorithm. CSPCA was designed to be as simple as the conventional SPCA algorithm. Unlike the conventional matrix techniques, CSPCA executes simply with repeated calculations that use class information. Moreover, unlike SPCA, CSPCA does not incur any computation costs for determining the number of eigenvectors, because that number is determined by the number of classes in the input dataset. The eigenvectors that are learned to span the eigenspace maximize the variance of the samples as well as the matrix-based algorithm PCA and SPCA. In addition, the number of the eigenvectors is as many as the number of classes from the input dataset. By this characteristic, parameter selection is not needed for determining the number of eigenvectors unlike SPCA algorithm. In our experimental results, the eigenspace learned by CSPCA algorithm was either more efficient or similar to the SPCA algorithm's eigenspace for classification. In addition, in some experiments, CSPCA acquired a smaller number of eigenvectors than SPCA. Our experiments confirm that CSPCA significantly improves recognition in spite of the smaller number of eigenvectors. Therefore, the CSPCA algorithm is more effective in terms of feature extraction for classification and computation costs.

However, in the case of some datasets that we used, CSPCA's recognition accuracy was not good. The reason in these cases is that the data are distributed uniformly and the number of eigenvectors is greater. We must therefore choose effective eigenvectors in order to avoid the wastage of memory. In our future work, we will study the eigenvector selection method and perform additional experiments.

Acknowledgment

This research was partly supported by JSPS KAKENHI (22500202).

References

[1] R. O. Duda and P. E. Hart, "Pattern Classification and Scene Analysis," John Wiley & Sons, 1973.

[2] I. Jolliffe, "Principal Component Analysis," New York, Springer, 1986.

[3] M. Partridge and R. Calvo, "Fast dimensionality reduction and simple PCA," *Intelligent Data Analysis*, vol. 2, no. 1. pp. 203-214, 1998.

[4] M. Fukumi and Y. Mitsukura, "Feature Generation by Simple-FLD," *Proc. of 9th International Conference on Knowledge-Based Intelligent Information & Engineering Systems*, Melboume, Australia, Sept. 2005.

[5] M. Fukumi and Y. Mitsukura, "A Simple Feature Generation Method Based on Fisher Linear Discriminant Analysis," *Proc. of IASTED Interantional Conference on Signal and Information Processing*, pp. 342-346, Honolulu, Hawaii, USA, 2005.

[6] M. Soriano, E. Marszalec and M. Pietikainen, "Color correction for face images under different illuminants by RGB eigenfaces," *Proc. of 2nd International Conference on Audio- and Video-based Biometric Person Authentication* (AVBPA '99), pp. 146-153, Washington DC, USA, 1999.

[7] T. Oyama, Y. Matsumura, S. Karungaru, Y. Mitsukura and M. Fukumi, "Construction of Wrist Motion Recognition System," Proc. of 2006 RISP International Workshop on Nonlinear Circuits and Signal Processing (NCSP '06), pp. 385-388, Hawaii, USA, Mar. 2006.

[8] T. Oyama, S. Karungaru, S. Tsuge, Y. Mitsukura and M. Fukumi, "Fast Incremental Algorithm of Simple Principal Component Analysis," IEEJ Trans. On Electronics, Information and Systems, vol. 129, no. 1, pp. 112-117, 2009.

[9] T. Oyama, S. Karungaru, S. Tsuge, Y. Mitsukura and M. Fukumi, "Wrist EMG Signals Identification using Neural Networks," *Proc. of 35th Annual Conference of the IEEE Industrial Electronics Society* (IECON 2009), pp. 4322-4326, Porto, Portugal, Nov. 2009.

[10] C. Zhong, Q. S. Hu, F. Yang and M. X. Yin, "Software Quality Prediction Method with Hybrid Applying Principal Components Analysis and Wavelet Neural Network and Genetic Algorithm," *International Journal of Digital Content Technology and its Applications*, vol. 5, no. 3, 2011.

[11] F. Zhang, X. Yue, D. Wang and L. Xi, "A Principal Components Weighted Real-valued Negative Selection Algorithm," *International Journal of Digital Content Technology and its Applications*, vol. 5, no. 6, 2011.

[12] A. Asuncion and D. J. Newman, "UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]." Irvine, CA: University of California, School of Information and Computer Science, 2007.

Yohei Takeuchi received the B.E. degree from the University of Tokushima in 2006 and the M.E. degree from Kobe University in 2008. He had worked for TOYOTA Motor Corporation as a researcher of advanced development for about 2 years. He is currently a doctoral course student with the Graduate School of Advanced Technology and Science, the University of Tokushima. His research interests include pattern recognition, neural networks and image processing. He is a student member of IEEE.

Momoyo Ito received the B.E., M.E., and Ph.D. degrees in Faculty of Engineering and Resource Science, Akita University, Akita, Japan, in 2005, 2007, and 2010. She has been an Assistant Professor at The University of Tokushima since 2010. Her research interests include intelligent image processing and human behavior analysis using image information.

Minoru Fukumi received the B.E. and M.E. degrees from the University of Tokushima, in 1984 and 1987, respectively, and the doctor degree from Kyoto University in 1996. Since 1987, he has been with the Department of Information Science and Intelligent Systems, University of Tokushima. In 2005, he became a Professor in the same department. He received the best paper awards from the SICE in 1995 and Research Institute of Signal Processing in 2011 in Japan, and best paper awards from some international conferences. His research interests include neural networks, evolutionary algorithms, image processing and human sensing. He is a member of the IEEE, SICE, IEEJ, IPSJ, RISP and IEICE.