SICE Annual Conference 2008
August 20-22, 2008, The University Electro-Communications, Japan

An Incremental Principal Component Analysis Based on Dynamic Accumulation Ratio

Seiichi Ozawa[1], Kazuya Matsumoto[1], Shaoning Pang[2], and Nikola Kasabov[2]

[1]Graduate School of Engineering, Kobe University, Hyogo, Japan
(E-mail: ozawasei@kobe-u.ac.jp)
[2]Knowledge Engineering & Discover Research Institute,
Auckland University of Technology, Auckland, New Zealand
(E-mail: spang@aut.ac.nz, nkasabov@aut.ac.nz)

Abstract: We have proposed an online feature extraction method called Chunk Incremental Principal Component Analysis (CIPCA) where a chunk of data is trained at a time to update an eigenspace model. This paper presents an extended version in which the threshold for accumulation ratio is adaptively determined so that the classification accuracy for validation data is always maximized. To define the validation set online, the prototypes are selected from given training samples by k-means clustering or nearest neighbor classifier. The experimental results show that the proposed CIPCA can update the threshold properly so as to maintain high classification accuracy.

Keywords: feature extraction, online incremental learning, pattern recognition, principal component analysis

1. INTRODUCTION

When constructing an adaptive recognition system, we should consider two types of incremental learning: one is the incremental learning of feature space and the other is that of classifier [1]. As for the feature extraction, Hall and Martin have proposed a method to update eigenvectors and eigenvalues in an incremental way called Incremental Principal Component Analysis (IPCA) [2]. To enhance the learning efficiency, IPCA was extended such that a chunk of training samples are trained at a time [3]. Ozawa et al. [4, 5] also have proposed this type of extended IPCA called Chunk IPCA (CIPCA). In CIPCA, the dimensional augmentation is judged based on the accumulation ratio, and a proper threshold value should be set to the accumulation ratio to obtain good classification accuracy. For this purpose, we have proposed a method to choose a threshold by applying the cross-validation technique to an initial training set. However, once the threshold is chosen, it is fixed over the entire learning stages. Obviously, the optimality for such a threshold is not ensured in the incremental environment where the distribution of given training samples could be changed over time.

In this paper, we propose two methods to choose proper threshold values adaptively in Chunk IPCA such that the classification accuracy for validation data is always maximized. In the first method, a validation data set to find an optimal threshold for accumulation ratio is dynamically updated by using k-means clustering. In the second method, a validation data set is obtained by selecting evenly from the classified and misclassified data.

This paper is organized as follows. Section 2 gives a quick review on the conventional IPCA algorithm in which an eigenspace model is updated one after another with a single data. Section 3 describes the extended version of IPCA called CIPCA in which a chunk of data is trained at one time. Then, Section 4 presents two methods to update an optimal threshold for accumulation ratio in CIPCA. In Section 5, the experiments using two UCI data sets are conducted and the effectiveness of updating the threshold is studied. Finally, Section 6 summarizes this work.

2. INCREMENTAL PRINCIPAL COMPONENT ANALYSIS

2.1 Original IPCA

Let us review the IPCA algorithm proposed by Hall and Martin [2] briefly.

Assume that N training samples $\boldsymbol{x}_i \in \mathcal{R}^n$ $(i = 1, \cdots, N)$ have been presented so far, and an eigenspace model $\Omega = (\bar{\boldsymbol{x}}, \boldsymbol{U}_k, \boldsymbol{\Lambda}_k, N)$ is constructed by calculating the eigenvectors and eigenvalues from the covariance matrix of $\boldsymbol{x}_i$, where $\bar{\boldsymbol{x}}$ is a mean vector of $\boldsymbol{x}_i$ $(i = 1, \cdots, N)$, $\boldsymbol{U}_k$ is an $n \times k$ matrix whose column vectors correspond to the eigenvectors, and $\boldsymbol{\Lambda}_k$ is a $k \times k$ matrix whose diagonal elements correspond to the eigenvalues. Here, k is the number of eigen-axes spanning the eigenspace (i.e., eigenspace dimensionality).

Now, assume that the $(N + 1)$th training sample $\boldsymbol{y} \in \mathcal{R}^n$ is given. The addition of this new sample will lead to the changes in both mean vector and covariance matrix; therefore, the eigenvectors and eigenvalues should also be updated. The new mean input vector $\bar{\boldsymbol{x}}'$ is easily obtained as follows:

$$\bar{\boldsymbol{x}}' = \frac{1}{N+1}(N\bar{\boldsymbol{x}} + \boldsymbol{y}) \in \mathcal{R}^n. \quad (1)$$

When updating the eigenspace model Ω, we need to check if the eigenspace should be enlarged in term of dimensionality. If the new sample includes almost all energy in the current eigenspace, the dimensionality does not need to be changed. However, if the eigenspace includes certain energy in the complementary eigenspace, the dimensional augmentation is needed, or crucial information on the new sample might be lost. In IPCA, the judgment of the eigenspace augmentation is made based on the norm of the following residue vector $\boldsymbol{h} \in \mathcal{R}^n$:

$$\boldsymbol{h} = (\boldsymbol{y} - \bar{\boldsymbol{x}}) - \boldsymbol{U}_k \boldsymbol{g} \tag{2}$$

where $\boldsymbol{g} = \boldsymbol{U}_k^T(\boldsymbol{y} - \bar{\boldsymbol{x}})$. Here, T means the transpose of vectors and matrices. When the norm of $\boldsymbol{h}$ is larger than a threshold value η, the dimensionality of the current eigenspace is increased from k to $k+1$, and a new eigen-axis is added in the direction of $\boldsymbol{h}$. Otherwise, the dimensionality of the eigenspace remains the same.

It has been shown that the eigenvectors and eigenvalues are updated by solving the following intermediate eigenproblem [2]:

$$\left(\frac{N}{N+1}\begin{bmatrix} \boldsymbol{\Lambda}_k & \boldsymbol{0} \\ \boldsymbol{0}^T & 0 \end{bmatrix} + \frac{N}{(N+1)^2}\begin{bmatrix} \boldsymbol{g}\boldsymbol{g}^T & \gamma\boldsymbol{g} \\ \gamma\boldsymbol{g}^T & \gamma^2 \end{bmatrix}\right) \times \boldsymbol{R} = \boldsymbol{R}\boldsymbol{\Lambda}'_{k+1} \tag{3}$$

where $\gamma = \tilde{\boldsymbol{h}}^T(\boldsymbol{y} - \bar{\boldsymbol{x}})$, $\boldsymbol{R}$ is a $(k+1) \times (k+1)$ matrix whose column vectors correspond to the eigenvectors obtained from the above intermediate eigenproblem, $\boldsymbol{\Lambda}'_{k+1}$ is the new eigenvalue matrix, and $\boldsymbol{0}$ is a k-dimensional zero vector. Using the solution $\boldsymbol{R}$, the new $n \times (k+1)$ eigenvector matrix $\boldsymbol{U}'_{k+1}$ is calculated as follows:

$$\boldsymbol{U}'_{k+1} = [\boldsymbol{U}_k,\ \hat{\boldsymbol{h}}]\boldsymbol{R} \tag{4}$$

where

$$\hat{\boldsymbol{h}} = \begin{cases} \boldsymbol{h}/\|\boldsymbol{h}\| & \text{if } \|\boldsymbol{h}\| > \eta \\ \boldsymbol{0} & \text{otherwise.} \end{cases} \tag{5}$$

Here, η is a small threshold value which is set to zero in the original IPCA [2]. From Eq. (4), intuitively we can consider that $\boldsymbol{R}$ gives a rotation from old eigen-axes to new ones; hence, let us call $\boldsymbol{R}$ *rotation matrix* here. Note that if $\hat{\boldsymbol{h}} = \boldsymbol{0}$ (i.e., the case that the dimensional augmentation was not needed), $\boldsymbol{R}$ degenerates into an $n \times k$ matrix, and $\boldsymbol{U}'_{k+1}$ and $\boldsymbol{\Lambda}'_{k+1}$ also degenerate to the $n \times k$ matrix and the $k \times k$ matrix, respectively.

2.2 A Criterion for Eigen-axes Augmentation

As seen from Eq. (5), a new eigen-axis is augmented whenever the norm of a residue vector is larger than a threshold value η in the original IPCA. However, this is not a good criterion in practice because a suitable threshold can be varied depending on the magnitude of input values. If the threshold is too small, the dimensionality of a feature space could be excessively large and an efficient feature space with small dimensions is hard to be constructed; this may deteriorate both generalization performance and computational efficiency. On the other hand, if the threshold is too large, essential information on training samples is lost unexpectedly.

To reduce the dependency of the threshold on input values, the following accumulation ratio is often used as a criterion:

$$A(\boldsymbol{U}_k) = \frac{\sum_{i=1}^{k}\lambda_i}{\sum_{i=1}^{n}\lambda_i} \tag{6}$$

where $\boldsymbol{U}_k = \{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_k\}$ is the eigenvector matrix whose column vectors span the k-dimensional feature space, λ_i $(i = 1, \cdots, k)$ is the eigenvalue of $\boldsymbol{u}_i$, and n is the dimensionality of the input space, respectively. By specifying an appropriate threshold value θ, the feature space dimensions are automatically determined by searching for a minimum k such that $A(\boldsymbol{U}_k) > \theta$ holds. In general, the update of Eq. (6) cannot be done without the training samples given previously. This is a serious problem when a one-pass incremental learning algorithm is developed. To overcome this problem, we need an incremental update algorithm of $A(\boldsymbol{U}_k)$ without keeping all the past training samples.

In [6], we derived the incremental update equation for $A'(\boldsymbol{U}_k)$ in Eq. (6), and it is given by

$$A'(\boldsymbol{U}_k) = \frac{N(N+1)\sum_{i=1}^{k}\lambda_i + N\|\boldsymbol{U}_k^T(\boldsymbol{y} - \bar{\boldsymbol{x}})\|^2}{N(N+1)\sum_{i=1}^{n}\lambda_i + N\|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2} \tag{7}$$

where $\boldsymbol{U}_k = \{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_k\}$. Note that no past sample is necessary for the incremental update of $A'(\boldsymbol{U}_k)$.

The eigen-axis augmentation is judged by using $A'(\boldsymbol{U}_k)$ as a criterion. Instead of Eq. (4), the update of $\boldsymbol{U}'$ is carried out based on the following equation.

$$\boldsymbol{U}'_{k+1} = [\boldsymbol{U}_k,\ \hat{\boldsymbol{h}}]\boldsymbol{R} \tag{8}$$

where

$$\hat{\boldsymbol{h}} = \begin{cases} \boldsymbol{h}/\|\boldsymbol{h}\| & \text{if } A(\boldsymbol{U}_k) < \theta \\ \boldsymbol{0} & \text{otherwise.} \end{cases} \tag{9}$$

Here, θ is a threshold value.

3. CHUNK IPCA

As stated in Section 1, the original IPCA is applied on one sample at a time, and the intermediate eigenproblem must be solved repeatedly for every training sample. Hence, the learning may get stuck in a deadlock if a large chunk of training samples is given to learn in a short term. To overcome this problem, we extend the original IPCA, so that the eigenspace model Ω can be updated with a chunk of training samples in a single operation. Let us call this extended algorithm *Chunk IPCA* [4, 5].

3.1 Update of Mean Vector and Eigen-axes

Let us assume that N training samples $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\} \in \mathcal{R}^{n \times N}$ have been given so far and they were already discarded. Instead of keeping actual training samples, we have an eigenspace model $\Omega = (\bar{\boldsymbol{x}}, \boldsymbol{U}_k, \boldsymbol{\Lambda}_k, N)$ where $\bar{\boldsymbol{x}}$, $\boldsymbol{U}_k$, and $\boldsymbol{\Lambda}_k$ are a mean input vector, an $n \times k$ eigenvector matrix, and a $k \times k$ eigenvalue matrix, respectively. Now, assume that a chunk of L training samples $\boldsymbol{Y} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_L\} \in \mathcal{R}^{n \times L}$ is presented.

Without the previous training samples $\boldsymbol{X}$, the updated mean vector $\bar{\boldsymbol{x}}'$ is easily obtained as follows:

$$\begin{aligned} \bar{\boldsymbol{x}}' &= \frac{1}{N+L}\left(\sum_{i=1}^{N}\boldsymbol{x}_i + \sum_{j=1}^{L}\boldsymbol{y}_j\right) \\ &= \frac{1}{N+L}(N\bar{\boldsymbol{x}} + L\bar{\boldsymbol{y}}). \end{aligned} \tag{10}$$

On the other hand, the updated covariance matrix is given in the following form [1]:

$$\begin{aligned} \boldsymbol{C}' &= \frac{1}{N+L}\Big[N\boldsymbol{C} + \frac{NL^2}{(N+L)^2}(\bar{\boldsymbol{y}}-\bar{\boldsymbol{x}})(\bar{\boldsymbol{y}}-\bar{\boldsymbol{x}})^T \\ &+\frac{N^2}{(N+L)^2}\sum_{i=1}^{L}(\boldsymbol{y}_i-\bar{\boldsymbol{x}})(\boldsymbol{y}_i-\bar{\boldsymbol{x}})^T \\ &+\frac{L(L+2N)}{(N+L)^2}\sum_{i=1}^{L}(\boldsymbol{y}_i-\bar{\boldsymbol{y}})(\boldsymbol{y}_i-\bar{\boldsymbol{y}})^T\Big]. \end{aligned} \quad (11)$$

Suppose that l eigen-axes must be augmented to avoid the serious loss of essential input information when a chunk of L training samples $\boldsymbol{Y}$ is provided; that is, the eigenspace dimensions are increased by l. Let us denote the augmented eigen-axes as follows:

$$\boldsymbol{H} = [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_l] \in \mathcal{R}^{n\times l}. \quad (12)$$

Then the updated eigenvector matrix $\boldsymbol{U}'_{k+l}$ is represented by using the rotation matrix $\boldsymbol{R}$ and the current eigenvector matrix $\boldsymbol{U}_k$.

$$\boldsymbol{U}'_{k+l} = [\boldsymbol{U}_k, \boldsymbol{H}]\boldsymbol{R}. \quad (13)$$

Hence a new eigenvalue problem to be solved is given by

$$\begin{aligned} &\boldsymbol{C}'\boldsymbol{U}'_{k+l} = \boldsymbol{U}'_{k+l}\boldsymbol{\Lambda}'_{k+l} \Rightarrow \\ &\qquad [\boldsymbol{U}_k, \boldsymbol{H}]^T\boldsymbol{C}'[\boldsymbol{U}_k, \boldsymbol{H}]\boldsymbol{R} = \boldsymbol{R}\boldsymbol{\Lambda}'_{k+l} \end{aligned} \quad (14)$$

where $\boldsymbol{\Lambda}'_{k+l}$ is a new eigenvalue matrix. Substituting Eqs. (11) and (12) into Eq. (14), the following intermediate eigenproblem for Chunk IPCA is obtained.

$$\begin{aligned} &\Big(\frac{N}{N+L}\begin{bmatrix}\boldsymbol{\Lambda}_k & \boldsymbol{0} \\ \boldsymbol{0}^T & 0\end{bmatrix} + \frac{NL^2}{(N+L)^3}\begin{bmatrix}\bar{\boldsymbol{g}}\bar{\boldsymbol{g}}^T & \bar{\boldsymbol{g}}\bar{\boldsymbol{\gamma}}^T \\ \bar{\boldsymbol{\gamma}}\bar{\boldsymbol{g}}^T & \bar{\boldsymbol{\gamma}}\bar{\boldsymbol{\gamma}}^T\end{bmatrix} \\ &+\frac{N^2}{(N+L)^3}\sum_{i=1}^{L}\begin{bmatrix}\boldsymbol{g}'_i\boldsymbol{g}'^T_i & \boldsymbol{g}'_i\boldsymbol{\gamma}'^T_i \\ \boldsymbol{\gamma}'_i\boldsymbol{g}'^T_i & \boldsymbol{\gamma}'_i\boldsymbol{\gamma}'^T_i\end{bmatrix} \\ &+\frac{L(L+2N)}{(N+L)^3}\sum_{i=1}^{L}\begin{bmatrix}\boldsymbol{g}''_i\boldsymbol{g}''^T_i & \boldsymbol{g}''_i\boldsymbol{\gamma}''^T_i \\ \boldsymbol{\gamma}''_i\boldsymbol{g}''^T_i & \boldsymbol{\gamma}''_i\boldsymbol{\gamma}''^T_i\end{bmatrix}\Big)\boldsymbol{R} \\ &= \boldsymbol{R}\boldsymbol{\Lambda}'_{k+l} \end{aligned} \quad (15)$$

where

$$\begin{aligned} \bar{\boldsymbol{g}} &= \boldsymbol{U}_k^T(\bar{\boldsymbol{y}}-\bar{\boldsymbol{x}}), & \boldsymbol{g}'_i &= \boldsymbol{U}_k^T(\boldsymbol{y}_i-\bar{\boldsymbol{x}}), \\ \boldsymbol{g}''_i &= \boldsymbol{U}_k^T(\boldsymbol{y}_i-\bar{\boldsymbol{y}}), & \bar{\boldsymbol{\gamma}} &= \boldsymbol{H}^T(\bar{\boldsymbol{y}}-\bar{\boldsymbol{x}}), \\ \boldsymbol{\gamma}'_i &= \boldsymbol{H}^T(\boldsymbol{y}_i-\bar{\boldsymbol{x}}), & \boldsymbol{\gamma}''_i &= \boldsymbol{H}^T(\boldsymbol{y}_i-\bar{\boldsymbol{y}}). \end{aligned}$$

Solving this intermediate eigenproblem, a new rotation matrix $\boldsymbol{R}$ and the eigenvalue matrix $\boldsymbol{\Lambda}'_{k+l}$ are obtained. Then, the corresponding new eigenvector matrix $\boldsymbol{U}'_{k+l}$ is given by Eq. (13).

3.2 Augmentation of Eigen-axes

In Chunk IPCA, the number of eigen-axes to be augmented is determined by finding a minimum k such that the accumulation ratio $A(\boldsymbol{U}_k)$ in Eq. (6) satisfies the following condition: $A(\boldsymbol{U}_k) \geq \theta$. However, the number of augmented eigen-axes is not restricted to one when a chunk of training samples is learned at a time. Therefore, the update equation of $A(\boldsymbol{U}_k)$ in Eq. (7) must be modified such that it can be updated with a chunk of training samples in one-pass.

The update equation for the accumulation ratio $A(\boldsymbol{U}_k)$ is given by

$$\begin{aligned} &A'(\boldsymbol{U}_k) \\ &= \frac{\sum_{i=1}^{k}\lambda_i + \frac{L}{N+L}\|\bar{\boldsymbol{g}}\|^2 + \frac{1}{N}\sum_{j=1}^{L}\|\boldsymbol{g}''_i\|^2}{\sum_{i=1}^{n}\lambda_i + \frac{L}{N+L}\|\bar{\boldsymbol{x}}-\bar{\boldsymbol{y}}\|^2 + \frac{1}{N}\sum_{j=1}^{L}\|\boldsymbol{y}_j-\bar{\boldsymbol{y}}\|^2}. \end{aligned} \quad (17)$$

As we can see from Eq. (17), no past sample is needed to update $A(\boldsymbol{U}_k)$.

In IPCA, a new eigen-axis is selected so as to be perpendicular to the existing eigenvectors which are given by the column vectors of $\boldsymbol{U}_k$. A straightforward way to get new eigen-axes is to apply Gram-Schmidt orthogonalization technique to the given chunk of training samples [3]. If the chunk samples are represented by $\tilde{L}$ linearly independent vectors, the maximum number of eigen-axes to be augmented is also $\tilde{L}$. However, the feature space spanned by all of the augmented eigen-axes is redundant in general; in addition, if the chunk size is large, the computation costs to solve the intermediate eigenproblem in Eq. (15) would be considerably expensive. Therefore, we need to select informative eigen-axes from the $\tilde{L}$ eigen-axes efficiently.

Algorithm 1 Eigen-axis Selection in CIPCA

```
Input:
    • Eigenspace model: $\Omega = (\bar{\boldsymbol{x}}, \boldsymbol{U}_k, \boldsymbol{\Lambda}_k, N)$.
    • Threshold for accumulation ratio: $\theta$.
    • Training samples: $\boldsymbol{y}_l \ (l = 1, \cdots, L)$.
  $\boldsymbol{H} = \{\ \}$.
  Calculate the accumulation ratio $A'(\boldsymbol{U}_k)$ in Eq. (17).
  if $A'(\boldsymbol{U}_k) \geq \theta$ then
    Terminate this algorithm.
  end if
  Obtain a set $\mathcal{L}$ of the following residue vectors $\boldsymbol{h}_l$:
      $\boldsymbol{h}_l = \frac{\boldsymbol{r}_l}{\|\boldsymbol{r}_l\|}$    (16)
  where
      $\boldsymbol{r}_l = (\boldsymbol{y}_l - \bar{\boldsymbol{x}}) - [\boldsymbol{U}_k, \boldsymbol{H}][\boldsymbol{U}_k, \boldsymbol{H}]^T(\boldsymbol{y}_l - \bar{\boldsymbol{x}})$.
  repeat
    Find the following residue vector $\boldsymbol{h}_{l^*}$ which gives
    the maximum accumulation ratio $A'([\boldsymbol{U}_k, \boldsymbol{H}, \boldsymbol{h}_l])$:
        $l^* = \arg\max_l A'([\boldsymbol{U}_k, \boldsymbol{H}, \boldsymbol{h}_l])$.
    Update $\boldsymbol{H}' = [\boldsymbol{H}, \boldsymbol{h}_{l^*}]$ and remove $\boldsymbol{h}_{l^*}$ from $\mathcal{L}$.
  until $A'([\boldsymbol{U}_k, \boldsymbol{H}']) < \theta$
Output: Augmented eigen-axes: $\boldsymbol{H}'$.
```

Algorithm 2 Generate validation data set in CIPCA-proto

```
Input:
    • The number of classes: $C$.
    • Initial training data set:
        $\boldsymbol{X}(0) = \{\boldsymbol{X}^1(0), \cdots \boldsymbol{X}^C(0)\}$.
    • $k$-means clustering method.
  // Initial Validation Set
  for $c = 1, \cdots, C$ do
    Apply an initial training data set $\boldsymbol{X}^c(0)$ of class $c$ to $k$-means clustering method.
    Obtain the $k$ prototype vectors:
      $\boldsymbol{P}^c(0) = \{\boldsymbol{p}_1^c(0), \cdots, \boldsymbol{p}_k^c(0)\}$.
  end for
  Define a whole set of $\boldsymbol{P}^c(0)$ $(c = 1, \cdots, C)$ as initial validation set $\boldsymbol{V}(0)$.
  // Update Validation Set
  $t = 1$
  repeat
    for $c = 1, \cdots, C$ do
      Apply a new training data set $\boldsymbol{X}^c(t)$ and the prototype set $\boldsymbol{P}^c(t-1)$ to $k$-means clustering method.
      Obtain the $k$ prototype vectors:
        $\boldsymbol{P}^c(t) = \{\boldsymbol{p}_1^c(t), \cdots, \boldsymbol{p}_k^c(t)\}$.
    end for
    Define a whole set of $\boldsymbol{P}^c(t)$ $(c = 1, \cdots, C)$ as validation set $\boldsymbol{V}(t)$.
    $t \leftarrow t + 1$.
  until No training data is given.
```

To construct a compact feature space, we should find a smallest set of augmented eigen-axes such that the eigenspace includes as much the energy of the given chunk data as possible. A straightforward way to find the set is to select an eigen-axis one by one, each of which gives a maximum accumulation ratio. The algorithm of the eigen-axis selection is summarized in Algorithm 1.

4. METHOD TO GENERATE VALIDATION DATA

In Chunk IPCA, the number of eigen-axes to be augmented is determined by finding a minimum k such that $A'(\boldsymbol{U}_k) \geq \theta$ holds where the accumulation ratio is calculated incrementally. To determine an optimal threshold θ, a cross-validation method can be applied to a validation data set at every several learning stages. That is to say, a threshold could be selected and changed over learning stages such that the classification accuracy is maximized for the validation data set.

In this paper, we propose two methods to generate a validation data set online such that the distribution of training examples is well approximated. In the first method, the validation data set is obtained by selecting k prototypes from the previous validation data and the given training samples. This can be done by applying the conventional k-means clustering to those data. In the second method, the validation data are selected so that a half of the data comes from easy data and the other half comes from tough data in terms of classification.

Algorithm 3 Generate validation data set in CIPCA-boosting

```
Input:
    • The number of classes $C$.
    • Initial training data set:
        $\boldsymbol{X}(0) = \{\boldsymbol{X}^1(0), \cdots \boldsymbol{X}^C(0)\}$.
    • Nearest neighbor (NN) classifier.
  // Initial Validation Set
  for $c = 1, \cdots, C$ do
    Classify an initial training data set $\boldsymbol{X}^c(0)$ of class $c$ with NN classifier based on the leave-one-out method.
    Separate $\boldsymbol{X}^c(0)$ into the two data sets which consist of the classified and misclassified data.
    Select the $k$ prototype vectors:
    $\boldsymbol{P}^c(0) = \{\boldsymbol{p}_1^c(0), \cdots, \boldsymbol{p}_k^c(0)\}$ from the two sets randomly.
  end for
  Define a whole set of $\boldsymbol{P}^c(0)$ $(c = 1, \cdots, C)$ as initial validation set $\boldsymbol{V}(0)$.
  // Update Validation Set
  $t = 1$
  repeat
    for $c = 1, \cdots, C$ do
      Classify a new training data set $\boldsymbol{X}^c(t)$ and the prototype set $\boldsymbol{P}^c(t-1)$ with NN classifier based on the leave-one-out method.
      Separate $\boldsymbol{X}^c(t)$ and $\boldsymbol{P}^c(t-1)$ into the two data sets which consist of the classified and misclassified data.
      Select the $k$ prototype vectors $\boldsymbol{P}^c(t) = \{\boldsymbol{p}_1^c(t), \cdots, \boldsymbol{p}_k^c(t)\}$ from the two sets randomly.
    end for
    Define a whole set of $\boldsymbol{P}^c(t)$ $(c = 1, \cdots, C)$ as validation set $\boldsymbol{V}(t)$.
    $t \leftarrow t + 1$.
  until No training data is given.
```

These methods to generate a validation data set are summarized in Algorithm 2 and 3.

5. EXPERIMENTS

We select two datasets for evaluation from the UCI Machine Learning Repository: Segmentation and Vowel-context data. The data information is shown in Table 1.

To construct an initial eigenspace, a part of training samples are applied to the conventional PCA. The rate of an initial dataset is set to 10% in all the experiments. The remaining 90% of training samples are sequentially provided to learn as shown in Fig. 1. The extended CIPCA works even though any size of data chunk is given at each learning stage. However, for the sake of simplicity, the chunk size is fixed at 20 over the entire learning period.

A chunk of training samples is randomly selected and

Table 1 Evaluated datasets.

Database	#Attrib.	#Classes	#Train.	#Test
Segment.	19	7	2,100	210
Vowel	10	11	528	462

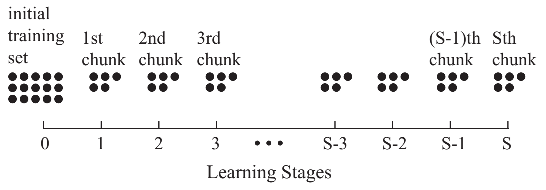


Fig. 1 The presentation of training samples in the assumed incremental learning environments.

it has no overlap with other chunks; hence all the training samples are presented only once because we assume one-pass incremental learning environments. Since the performance of incremental learning generally depends on the sequence of training samples, 50 trials with different sequences of training samples are carried out to evaluate the average performance for the test datasets in Table 1.

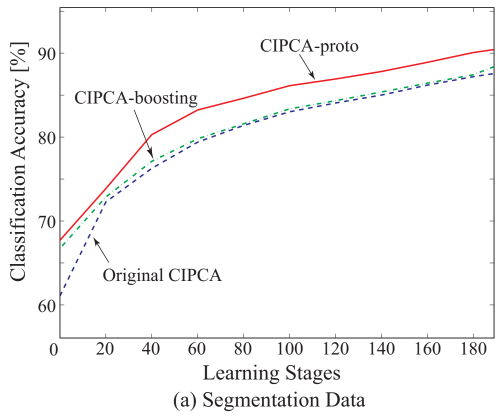
The evolutions of classification accuracy are shown in Figs.2 (a) and (b). In Fig.2, "CIPCA-proto" and "CIPCA-boost" correspond to the proposed CIPCA methods where a validation data set is dynamically updated by k-means clustering and classification, respectively.

In the original CIPCA, the threshold value is determined at the initial leaning stage and it is fixed over a learning session. As seen from Fig. 2 (a), the classification accuracy of the proposed CIPCA-proto for Segmentation data is largely improved as compared with the other two methods. For Vowel data, although there is no significant difference in the final accuracy of the three methods, the accuracies of CIPCA-proto and CIPCA-boosting are quickly going up as compared with the original CIPCA. Therefore, one can conclude that the online performance of CIPCA-proto is better than those of CIPCA-boosting and the original CIPCA for the two UCI datasets.

6. CONCLUSIONS

In this paper, we proposed the two methods to determine the threshold for accumulation ratio adaptively in Chunk IPCA such that the classification accuracy for validation data is always maximized. In the first method, a validation data set to find an optimal accumulation ratio is dynamically updated by using k-means clustering. On the other hand, in the second method, a validation data set is obtained by selecting evenly from the classified and misclassified data. These methods are called CIPCA-proto and CIPCA-boosting, respectively.

To evaluate the proposed methods, the experiments using two UCI data sets are conducted and the effectiveness of updating the threshold is studied. The experimental results show that the extended Chunk IPCA can determine a proper threshold dynamically so as to keep high classification accuracy. For Segmentation data, CIPCA-proto outperforms both CIPCA-boosting and the original CIPCA in classification accuracy.



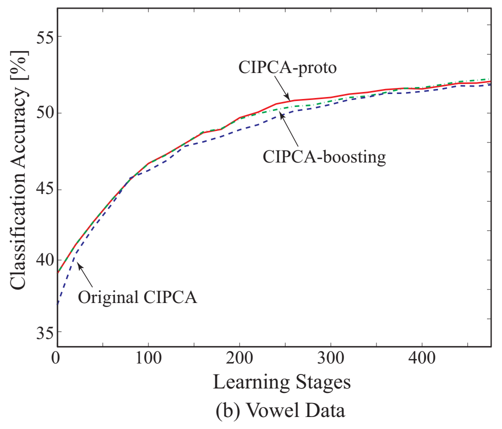


Fig. 2 Time evolution of classification accuracy for three CIPCA algorithms.

REFERENCES

[1] S. Ozawa, S. L. Toh, S. Abe, S. Pang and N. Kasabov, "Incremental Learning of Feature Space and Classifier for Face Recognition," *Neural Networks*, vol. 18, nos. 5-6, pp. 575-584, 2005.

[2] P. Hall and R. Martin, "Incremental Eigenanalysis for Classification," *Proc. of British Machine Vision Conference*, vol. 1, pp. 286-295, 1998.

[3] P. Hall, D. Marshall, and R. Martin, "Merging and Splitting Eigenspace Models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 9, pp. 1042-1049, 2000.

[4] S. Ozawa, S. Pang, and N. Kasabov, "An Incremental Principal Component Analysis for Chunk Data," *Proc. of FUZZ-IEEE*, pp. 10493-10500, 2006.

[5] S. Ozawa, S. Pang, and N. Kasabov, "Incremental Learning of Chunk Data for On-line Pattern Classification Systems," *IEEE Trans. on Neural Networks* (in press).

[6] S. Ozawa, S. Pang, and N. Kasabov, "A Modified Incremental Principal Component Analysis for On-line Learning of Feature Space and Classifier," in C. Zhang, H. W. Guesgen, and W. K. Yeap (Eds.), *PRICAI 2004: Trends in Artificial Intelligence* LNAI, Springer-Verlag, pp. 231-240, 2004.