Accepted for IEEE TKDE (TKDE113771.2)

Feature Extraction Based on ICA for Binary Classification Problems

Nojun Kwak and Chong-Ho Choi

{triplea|chchoi}@csl.snu.ac.kr

Phone:(+82-2)880-7310, Fax:(+82-2)885-4459

School of Electrical Engineering and Computer Science,

Seoul National University

San 56-1, Shinlim-dong, Kwanak-ku, Seoul 151-742 KOREA

Nojun Kwak is a Ph.D. student in the School of Electrical Engineering and Computer Science, Seoul National University, Seoul, Korea.

Chong-Ho Choi is with the School of Electrical Engineering and Computer Science, and also with the Automation and Systems Research Institute, Seoul National University, Seoul, Korea.

The corresponding author is Nojun Kwak and his e-mail address is underlined.

This work is partially supported by the Brain Science and Engineering Program of the Korea Ministry of Science and Technology.

August 12, 2002

Abstract

In manipulating data such as in supervised learning, we often extract new features from the original features for the purpose of reducing the dimensions of feature space and achieving better performance. In this paper, we show how standard algorithms for independent component analysis (ICA) can be appended with binary class labels to produce a number of features that do not carry information about the class labels – these features will be discarded – and a number of features that do. We also provide a local stability analysis of the proposed algorithm. The advantage is that general ICA algorithms become available to a task of feature extraction for classification problems by maximizing the joint mutual information between class labels and new features, although only for two-class problems. Using the new features, we can greatly reduce the dimension of feature space without degrading the performance of classifying systems.

Keywords

Feature extraction, ICA, stability, classification.

I. Introduction

In supervised learning, one is given an array of attributes to predict the target value or output class. These attributes are called *features*, and there may exist irrelevant or redundant features to complicate the learning process, thus leading to incorrect prediction. Even when the features presented contain enough information about the output class, they may not predict the output correctly because the dimension of feature space may be so large that it may require numerous instances to determine the relationship. This problem is commonly referred to as the *curse of dimensionality* [1]. Some experiments have also reported that the performance of classifier systems deteriorates as new irrelevant features are added [2]. Though some of the modern classifiers, such as support vector machine (SVM), are surprisingly tolerant to extra irrelevant information, these problems can be avoided by selecting only the relevant features or extracting new features containing the maximal information about the class label from the original ones. The former methodology is called feature selection or subset selection, while the latter is named feature extraction which includes all the methods that compute any functions, logical or numerical.

This paper considers the feature extraction problem since it often results in improved performance by extracting new features which are arbitrary linear combinations of original features, especially when small dimensions are required.

Though the principal component analysis (PCA) is the most popular [3], by its nature, it is not well-fitted for supervised learning since it does not make use of any output class information in deciding the principal components. The main drawback of this method is that the extracted features are not invariant under transformation. Merely scaling the attributes changes resulting features.

Unlike PCA, Fisher's linear discriminant analysis (LDA) [4] focuses on classification problems to find optimal linear discriminating functions. Though it is a very simple and powerful method for feature extraction, the application of this method is limited to the case in which classes have significant differences between means, since it is based on the information about the differences between means.

Another common method of feature extraction is to use a feedforward neural network such as multilayer perceptron (MLP). This method uses the fact that in the feedforward structure the output class is determined through the hidden nodes which produce transformed forms of original input features. This notion can be understood as squeezing the data through a bottleneck of a few hidden units. Thus, the hidden node activations are interpreted as new features in this approach. This line of research includes [5] - [9]. Fractal encoding [10] and wavelet transformation [11] have also been used for feature extraction.

Recently, in neural networks and signal processing circles, independent component analysis (ICA), which was devised for blind source separation problems, has received a great deal of attention because of its potential applications in various areas. Bell and Sejnowski [12] have developed an unsupervised learning algorithm performing ICA based on entropy maximization in a single-layer feedforward neural network. ICA can be very useful as a dimension-preserving transform because it produces statistically independent components, and some have directly used ICA for feature extraction and selection [13] - [16]. Recent research [17], [18] is focused on extraction of features relevant to task based on mutual information maximization methods. In this research, Renyi's entropy measure was used instead of that of Shannon.

In this paper, we show how standard algorithms for ICA can be appended with binary class labels to produce a number of features that do not carry information about the class label – these features will be discarded – and a number of features that do. The

advantage is that general ICA algorithms become available to a task of feature extraction by maximizing the joint mutual information between class labels and new features, although limited only for two-class problems. It is an extended version of [19] and this method is well-suited for classification problems. The proposed algorithm greatly reduces the dimension of feature space while improving classification performance.

This paper is organized as follows. In Section II, we briefly review some aspects of ICA. In Section III, we propose a new feature extraction algorithm and present a local stability analysis of the algorithm. In Section IV, we give some simulation results showing the advantages of the proposed algorithm. Conclusions follow in Section V.

II. Review of ICA

The problem of linear independent component analysis for blind source separation was developed in the literature [20] - [22]. In parallel, Bell and Sejnowski [12] have developed an unsupervised learning algorithm based on entropy maximization of a feedforward neural network's output layer, which is referred to as the Infomax algorithm. The Infomax approach, maximum likelihood estimation (MLE) approach, and negentropy maximization approach were shown to lead to identical methods [23] - [25].

The problem setting of ICA is as follows. Assume that there is an L-dimensional zero-mean non-Gaussian source vector $\boldsymbol{s}(t) = [s_1(t), \cdots, s_L(t)]^T$, such that the components $s_i(t)$'s are mutually independent, and an observed data vector $\boldsymbol{x}(t) = [x_1(t), \cdots, x_N(t)]^T$ is composed of linear combinations of sources $s_i(t)$ at each time point t, such that

$$\boldsymbol{x}(t) = A\boldsymbol{s}(t) \tag{1}$$

where A is a full rank $N \times L$ matrix with $L \leq N$. The goal of ICA is to find a linear mapping W such that each component of an estimate $\boldsymbol{u}$ of the source vector

$$\boldsymbol{u}(t) = W\boldsymbol{x}(t) = WA\boldsymbol{s}(t) \tag{2}$$

is as independent as possible. The original sources $\boldsymbol{s}(t)$ are exactly recovered when W is the pseudo-inverse of A up to some scale changes and permutations. For a derivation of an ICA algorithm, one usually assumes that $L = N$, because we have no idea about the

number of sources. In addition, sources are assumed to be independent of time t and are drawn from independent identical distribution $p_i(s_i)$.

Bell and Sejnowski [12] have used a feed-forward neural processor to develop the Infomax algorithm, one of the popular algorithms for ICA. The overall structure of the Infomax is shown in Fig. 1. This neural processor takes $\boldsymbol{x}$ as an input vector. The weight W is multiplied to the input $\boldsymbol{x}$ to give $\boldsymbol{u}$ and each component u_i goes through a bounded invertible monotonic nonlinear function $g_i(\cdot)$ to match the cumulative distribution of the sources. Let $y_i = g_i(u_i)$ as shown in the figure.

From the view of information theory, maximizing the statistical independence among variables u_i's is equivalent to minimizing mutual information among u_i's. This can be achieved by minimizing mutual information between y_i's, since the nonlinear transfer function $g_i(\cdot)$ does not introduce any dependencies.

In [12], it has been shown that by maximizing the joint entropy $H(\boldsymbol{y})$ of the output $\boldsymbol{y} = [y_1, \cdots, y_N]^T$ of a processor, we can approximately minimize the mutual information among the output components y_i's

$$I(\boldsymbol{y}) = \int p(\boldsymbol{y}) \log \frac{p(\boldsymbol{y})}{\prod_{i=1}^{N} p_i(y_i)} \, d\boldsymbol{y}. \tag{3}$$

Here, $p(\boldsymbol{y})$ is the joint probability density function (pdf) of a vector $\boldsymbol{y}$, and $p_i(y_i)$ is the marginal pdf of the variable y_i.

The joint entropy of the outputs of this processor is

$$\begin{aligned} H(\boldsymbol{y}) &= -\int p(\boldsymbol{y}) \log p(\boldsymbol{y}) d\boldsymbol{y} \\ &= -\int p(\boldsymbol{x}) \frac{p(\boldsymbol{x})}{\log |\det J(\boldsymbol{x})|} d\boldsymbol{x} \end{aligned} \tag{4}$$

where $J(\boldsymbol{x})$ is the Jacobian matrix whose (i,j)th element is partial derivative $\partial y_j / \partial x_i$. Note that $J(\boldsymbol{x}) = W$. Differentiating $H(\boldsymbol{y})$ with respect to W leads to the learning rule for ICA:

$$\Delta W \propto W^{-T} - \boldsymbol{\varphi}(\boldsymbol{u}) \boldsymbol{x}^T. \tag{5}$$

By multiplying $W^T W$ on the right, we get the natural gradient [26] speeding up the convergence rate

$$\Delta W \propto [I - \boldsymbol{\varphi}(\boldsymbol{u}) u^T] W \tag{6}$$

where

$$\boldsymbol{\varphi}(\boldsymbol{u}) = \left[-\frac{\frac{\partial p_1(u_1)}{\partial u_1}}{p_1(u_1)}, \cdots, -\frac{\frac{\partial p_N(u_N)}{\partial u_N}}{p_N(u_N)} \right]^T . \tag{7}$$

The parametric density estimation $p_i(u_i)$ plays an important role in the success of the learning rule in (6). If we assume $p_i(u_i)$ be Gaussian, $\varphi_i(u_i) = -\dot{p}_i(u_i)/p_i(u_i)$ becomes a linear function of u_i with a positive coefficient and the learning rule (6) becomes unstable. Also note that the sum of Gaussians is a Gaussian, and thus with given observations $\boldsymbol{x}$ which are mixtures of sources $\boldsymbol{s}$, the sources cannot be separated by any density related criterion if we assume $\boldsymbol{s}$ to be Gaussian. This is why we assume non-Gaussian sources.

There is a close relation between the assumption on the source distribution and the choice of the nonlinear function $g_i(\cdot)$. By simple computation with (3) and (4), the joint entropy $H(\boldsymbol{y})$ becomes

$$H(\boldsymbol{y}) = \sum_{i=1}^{N} H(y_i) - I(\boldsymbol{y}). \tag{8}$$

The maximal value for $H(\boldsymbol{y})$ is achieved when the mutual information among the outputs is zero and their marginal distributions are uniform. For a uniform distribution of y_i the distribution of u_i must be

$$p_i(u_i) = \left| \frac{\partial g_i(u_i)}{\partial u_i} \right| \tag{9}$$

because the relation between the pdf of y_i and that of u_i is

$$p_i(y_i) = p_i(u_i) / \left| \frac{\partial g_i(u_i)}{\partial u_i} \right|, \quad \text{for } p_i(y_i) \neq 0. \tag{10}$$

By the relationship (9), the estimate u_i of the source has a distribution that is approximately the form of the derivative of the nonlinearity.

Note that if we use the sigmoid function for $g_i(\cdot)$ as in [12], $p_i(u_i)$ in (9) becomes super-Gaussian, which has longer tails than the Gaussian pdf. Some researches [27], [26], [28] relax the assumption on the source distribution to be sub-Gaussian or super-Gaussian and [26] leads to the extended Infomax learning rule:

$$\Delta W \propto [I - D \tanh(\boldsymbol{u})\boldsymbol{u}^T - \boldsymbol{u}\boldsymbol{u}^T]W \tag{11}$$

$$\begin{cases} d_i = 1 & : \text{super-Gaussian} \\ d_i = -1 & : \text{sub-Gaussian.} \end{cases}$$

Here d_i is the ith element of the N-dimensional diagonal matrix D, and it switches between sub- and super-Gaussian using the stability analysis.

In this paper, we adopt the extended Infomax algorithm in [26] because it is easy to implement with less strict assumptions on source distribution.

III. Feature extraction based on ICA

ICA outputs a set of maximally independent vectors which are linear combinations of observed data. Although these vectors may find some applications in such areas as blind source separation [12] and data visualization [13], it does not fit for feature extraction for classification problems, because it is an unsupervised learning that does not use class information. In this section, we will propose a feature extraction algorithm for the classification problem by incorporating standard ICA algorithms with binary class labels.

The main idea of the proposed feature extraction algorithm is simple. In applying standard ICA algorithms to feature extraction for classification problems, it makes use of the binary class labels to produce two sets of new features; one that does not carry information about the class label (these features will be discarded) and the other that does (these will be useful for classification). The advantage is that general ICA algorithms become available to a task of feature extraction by maximizing the joint mutual information between class labels and new features, although only for two-class problems.

Before we present our algorithm ICA-FX (feature extraction algorithm based on ICA), we formalize the purpose of feature extraction.

A. Purpose

The success of a feature extraction algorithm depends critically on how much information about the output class is contained in the newly generated features.

Suppose that there are N normalized input features $\boldsymbol{x} = [x_1, \cdots, x_N]^T$ and a binary output class $c \in \{-1, 1\}$. Our purpose of the feature extraction is to extract $M(\leq N)$ new features $\boldsymbol{f}_a = [f_1, \cdots, f_M]^T$ from $\boldsymbol{x}$ containing maximal information of the class.

A useful lemma in relation to this is Fano's inequality [29] in information theory.

Lemma 1: (Fano's inequality) Let $\boldsymbol{f_a}$ and c be random variables which represent input features and output class, respectively. If we are to estimate the output class c using the

input features $\boldsymbol{f_a}$, the lower bound of error probability P_E satisfies the following inequality:

$$P_E \geq \frac{H(c|\boldsymbol{f_a}) - 1}{\log N_c} = \frac{H(c) - I(\boldsymbol{f_a}; c) - 1}{\log N_c} \tag{12}$$

where $H(\cdot)$, $H(\,\cdot\,|\,\cdot\,)$, and $I(\,\cdot\,;\,\cdot\,)$ are entropy, conditional entropy, and mutual information, respectively, and N_c is the number of classes.

Because the entropy of class $H(c)$ and the number of classes N_c is fixed, the lower bound of P_E is minimized when $I(\boldsymbol{f_a}; c)$ becomes maximum. Thus it is necessary for good feature extraction methods to extract features maximizing mutual information with the output class. But there is no transformation $T(\cdot)$ that can increase the mutual information between input features and output class as shown by the following data processing inequality [29].

Lemma 2: (Data processing inequality) Let $\boldsymbol{x}$ and c be random variables that represent input features and output class, respectively. For any deterministic function $T(\cdot)$ of $\boldsymbol{x}$, the mutual information between $T(\boldsymbol{x})$ and output class c is upper-bounded by the mutual information between $\boldsymbol{x}$ and c:

$$I(T(\boldsymbol{x}); c) \leq I(\boldsymbol{x}; c) \tag{13}$$

where the equality holds if the transformation is invertible.

Thus, the purpose of a feature extraction is to extract $M(\leq N)$ features $\boldsymbol{f}_a$ from $\boldsymbol{x}$, such that $I(\boldsymbol{f}_a; c)$, the mutual information between newly extracted features $\boldsymbol{f}_a$ and output class c, becomes as close as to $I(\boldsymbol{x}; c)$, the mutual information between original features $\boldsymbol{x}$ and output class c.

B. Algorithm : ICA-FX

In this subsection, we propose a feature extraction method by modifying a standard ICA algorithm for the purpose presented in the previous subsection. The main idea of the proposed method is to incorporate the binary class labels into the structure of standard ICA to extract a set of new features that provide information about class labels, as LDA does but using a method other than orthogonal projection.

Consider the structure shown in Fig. 2. Here, the original feature vector $\boldsymbol{x} = [x_1, \cdots, x_N]^T$ is fully connected to $\boldsymbol{u} = [u_1, \cdots, u_N]$, class label c is connected to $\boldsymbol{u}_a = [u_1, \cdots, u_M]$, and

$u_{N+1} = c$. In the figure, the weight matrix $\boldsymbol{W} \in \Re^{(N+1)\times(N+1)}$ becomes

$$\boldsymbol{W} = \begin{pmatrix} w_{1,1} & \cdots & w_{1,N} & w_{1,N+1} \\ \vdots & & \vdots & \vdots \\ w_{M,1} & \cdots & w_{M,N} & w_{M,N+1} \\ w_{M+1,1} & \cdots & w_{M+1,N} & 0 \\ \vdots & & \vdots & \vdots \\ w_{N,1} & \cdots & w_{N,N} & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}. \tag{14}$$

And let us denote the upper left $N \times N$ matrix of $\boldsymbol{W}$ as W.

Now our aim is to separate the input feature space $\boldsymbol{x}$ into two linear subspaces: one that is spanned by $\boldsymbol{f}_a = [f_1, \cdots, f_M]^T$ that contains maximal information about the class label c, and the other spanned by $\boldsymbol{f}_b = [f_{M+1}, \cdots, f_N]^T$ that is independent of c as much as possible.

The condition for this separation can be derived as follows. If we assume that the weight matrix $\boldsymbol{W}$ is nonsingular, we can see that $\boldsymbol{x}$ and $\boldsymbol{f} = [f_1, \cdots, f_N]^T$ span the same linear space and it can be represented with direct sum of $\boldsymbol{f}_a$ and $\boldsymbol{f}_b$. Then by Lemma 2, we can see that

$$\begin{aligned} I(\boldsymbol{x}; c) =& I(W\boldsymbol{x}; c) \\ =& I(\boldsymbol{f}; c) \\ =& I(\boldsymbol{f}_a, \boldsymbol{f}_b; c) \\ \geq& I(\boldsymbol{f}_a; c). \end{aligned} \tag{15}$$

The first equality holds because W is nonsingular and in the inequality on the last line, equality holds if $I(\boldsymbol{f}_b; c) = I(u_{M+1}, \cdots, u_N; c) = 0$.

If this is possible, we can reduce the dimension of input feature space from N to $M(< N)$ by using only $\boldsymbol{f}_a$ instead of $\boldsymbol{x}$, without losing any information about the target class.

To solve this problem, we interpret the feature extraction problem in the structure of the blind source separation (BSS) problem in the following.

(Mixing) Assume that there exist N independent non-Gaussian sources $\boldsymbol{s} = [s_1, \cdots, s_N]^T$ which are also independent of class label c. Assume also that the observed feature vector

$\boldsymbol{x}$ is the linear combination of the sources $\boldsymbol{s}$ and c with the mixing matrix $A \in \Re^{N\times N}$ and $\boldsymbol{b} \in \Re^{N\times 1}$; i.e.,

$$\boldsymbol{x} = A\boldsymbol{s} + \boldsymbol{b}c. \tag{16}$$

(Unmixing) Our unmixing stage is a little different from the BSS problem as shown in Fig. 2. Let us denote the last column of $\boldsymbol{W}$ without the $(N+1)$th element as $\boldsymbol{v} \in \Re^{N\times 1}$. Then the unmixing equation becomes

$$\boldsymbol{u} = W\boldsymbol{x} + \boldsymbol{v}c. \tag{17}$$

Suppose we have made $\boldsymbol{u}$ somehow equal to $\boldsymbol{e}$, the scaled and permuted version of source $\boldsymbol{s}$; i.e.,

$$\boldsymbol{e} \triangleq \Lambda\Pi\boldsymbol{s} \tag{18}$$

where Λ is a diagonal matrix corresponding to an appropriate scale and Π is a permutation matrix. Then, u_i's $(i = 1, \cdots, N)$ are independent of class c, and among the elements of $\boldsymbol{f} = W\boldsymbol{x}(= \boldsymbol{u} - \boldsymbol{v}c)$, $\boldsymbol{f}_b = [f_{M+1}, \cdots, f_N]^T$ will be independent of c because $v_i = w_{i,N+1} = 0$ for $i = M+1, \cdots, N$. Thus, we can extract $M(< N)$ dimensional new feature vector $\boldsymbol{f}_a$ by a linear transformation of $\boldsymbol{x}$ containing the maximal information about the class if the relation $\boldsymbol{u} = \boldsymbol{e}$ holds.

Now that the feature extraction problem is set in a similar form as the standard BSS or ICA problem, we can derive a learning rule for $\boldsymbol{W}$, using the the similar approach for the derivation of a learning rule for ICA. Because the Infomax approach, MLE approach, and negentropy maximization approach were shown to lead to the identical learning rule for ICA problems, as mentioned in the previous section, any approach can be used for the derivation. In this paper, we use MLE to obtain a learning rule.

If we assume that $\boldsymbol{u} = [u_1, \cdots, u_N]^T$ is a linear combination of the source $\boldsymbol{s}$; i.e., it is made to be equal to $\boldsymbol{e}$, a scaled and permutated version of the source $\boldsymbol{s}$ as in (18), and that each element of $\boldsymbol{u}$ is independent of other elements of $\boldsymbol{u}$ and it is also independent of class c, the log likelihood of the given data becomes

$$L(\boldsymbol{u}, c, \boldsymbol{W}) = \log|\det \boldsymbol{W}| + \sum_{i=1}^{N} \log p_i(u_i) + \log p(c) \tag{19}$$

because

$$p(\boldsymbol{x}, c) = |\det \boldsymbol{W}|\ p(\boldsymbol{u}, c) = |\det \boldsymbol{W}|\ \prod_{i=1}^{N} p_i(u_i)\ p(c). \tag{20}$$

Now, we are to maximize L, and this can be achieved by the steepest ascent method. Because the last term in (19) is a constant, differentiating (19) with respect to $\boldsymbol{W}$ leads to

$$\begin{aligned} \frac{\partial L}{\partial w_{i,j}} &= \frac{adj(w_{j,i})}{|\det \boldsymbol{W}|} - \varphi_i(u_i) x_j \qquad 1 \le i, j \le N \\ \frac{\partial L}{\partial w_{i,N+1}} &= -\varphi_i(u_i) c \qquad\qquad\qquad 1 \le i \le M \end{aligned} \tag{21}$$

where $adj(\cdot)$ is adjoint and $\varphi_i(u_i) = -\frac{dp_i(u_i)}{du_i}/p_i(u_i)$. Note that c has binary numerical values corresponding to the two categories.

We can see that $|\det \boldsymbol{W}| = |\det W|$ and $adj(w_{j,i})/|\det \boldsymbol{W}| = W_{i,j}^{-T}$. Thus the learning rule becomes

$$\begin{aligned} \Delta W &\propto\ W^{-T} - \boldsymbol{\varphi}(\boldsymbol{u}) \boldsymbol{x}^T \\ \Delta \boldsymbol{v}_a &\propto\ -\boldsymbol{\varphi}(\boldsymbol{u}_a) c. \end{aligned} \tag{22}$$

Since the two terms in (22) have different tasks regarding the update of separate matrices W and W_{N+1}, we can divide the learning process, and applying natural gradient on updating W, we get

$$\begin{aligned} W^{(t+1)} &= W^{(t)} + \mu_1 [I_N - \boldsymbol{\varphi}(\boldsymbol{u}) \boldsymbol{f}^T] W^{(t)} \\ \boldsymbol{v}_a^{(t+1)} &= \boldsymbol{v}_a^{(t)} - \mu_2 \boldsymbol{\varphi}(\boldsymbol{u}_a) c. \end{aligned} \tag{23}$$

Here $\boldsymbol{v}_a \triangleq [w_{1,N+1}, \cdots, w_{M,N+1}]^T \in \Re^M$, $\boldsymbol{\varphi}(\boldsymbol{u}) \triangleq [\varphi_1(u_1), \cdots, \varphi_N(u_N)]^T$, $\boldsymbol{\varphi}(\boldsymbol{u}_a) \triangleq [\varphi_1(u_1), \cdots, \varphi_M(u_M)]^T$, I_N is a $N \times N$ identity matrix, and μ_1 and μ_2 are learning rates that can be set differently. By this updating rule, the assumption that u_i's are independent of one another and of c will most be likely fulfilled by the resulting u_i's.

Note that the learning rule for W is the same as the original ICA learning rule [12], and also note that $\boldsymbol{f}_a$ corresponds to the first M elements of $W\boldsymbol{x}$. Therefore, we can extract the optimal features $\boldsymbol{f}_a$ by the proposed algorithm when it finds the optimal solution for W by (23).

C. Stability of ICA-FX

In this part, we will present the conditions of local stability of the ICA-FX algorithm. The local stability analysis in this paper undergoes almost the same procedure as that of general ICA algorithms in [30].

C.1 Stationary points

To begin with, let us first investigate the stationary point of the learning rule given in (23). Let us define

$$A_\star \triangleq A(\Lambda\Pi)^{-1}. \tag{24}$$

Now assuming that the output $\boldsymbol{u}$ is made to be equal to $\boldsymbol{e}$, then (16), (17), and (18) become

$$\begin{aligned} \boldsymbol{x} &= A_\star \boldsymbol{e} + \boldsymbol{b}c \\ \boldsymbol{e} &= W\boldsymbol{x} + \boldsymbol{v}c \end{aligned} \tag{25}$$

and we get

$$(I_N - WA_\star)\boldsymbol{e} = (W\boldsymbol{b} + \boldsymbol{v})c. \tag{26}$$

Because c and $\boldsymbol{e}$ are assumed to be independent of each other, W and $\boldsymbol{v}$ must satisfy

$$\begin{aligned} W &= A_\star^{-1} = \Lambda\Pi A^{-1} \\ \boldsymbol{v} &= -W\boldsymbol{b} = -A_\star^{-1}\boldsymbol{b} = -\Lambda\Pi A^{-1}\boldsymbol{b} \end{aligned} \tag{27}$$

if $\boldsymbol{u}$ were made to be equal to $\boldsymbol{e}$. This solution is a stationary point of learning rule (23) by the following theorem.

Theorem 1: The W and $\boldsymbol{v}$ satisfying (27) is a stationary point of the learning rule (23), and the scaling matrix Λ is uniquely determined up to a sign change in each component.

Proof: See Appendix I. ■

In most cases, we use odd increasing activation functions φ_i for ICA, and if we do the same for the ICA-FX, we can get the unique scale up to a sign and W and $\boldsymbol{v}$ in (27) is a stationary point.

C.2 Local asymptotic stability

Now let us investigate the condition for the stability of the stationary point given in (27). In doing so we introduce a new version of weight matrix Z and a set of scalars k_i's

such that

$$\begin{aligned} W^{(t)} &= Z^{(t)} W^* \\ v_i^{(t)} &= k_i^{(t)} v_i^* \ (\neq 0), \qquad 1 \le i \le M \end{aligned} \tag{28}$$

to follow the same procedure as in [30]. Here W^* and v_i^* are the optimal values of W and v_i which are $A_\star^{-1}$ and $-(A_\star^{-1}\boldsymbol{b})_i$, respectively. Note that the stability of W and v_i in the vicinity of W^* and v_i^* is equivalent to the stability of Z and k_i in the vicinity of the identity matrix I_N and 1.

If we multiply W^{*-1} to both sides of the learning rule for W in (23), we get

$$Z^{(t+1)} = \{I_N - \mu_1 G(Z^{(t)}, \boldsymbol{k}^{(t)})\} Z^{(t)} \tag{29}$$

where the (i,j)th element of $G \in \Re^{N\times N}$ is

$$\begin{aligned} G(Z^{(t)}, \boldsymbol{k}^{(t)})_{ij} &= \varphi_i(u_i) f_j - \delta_{ij} \\ &= \begin{cases} \varphi_i((Z^{(t)} W^* \boldsymbol{x})_i + k_i^{(t)} v_i^* c)(Z^{(t)} W^* \boldsymbol{x})_j - \delta_{ij} & \text{if } \ 1 \le i \le M \\ \varphi_i((Z^{(t)} W^* \boldsymbol{x})_i)(Z^{(t)} W^* \boldsymbol{x})_j - \delta_{ij} & \text{if } \ M < i \le N \end{cases}. \end{aligned} \tag{30}$$

Here, we denote $\boldsymbol{k} = [k_1, \cdots, k_M]^T$ for convenience.

In the learning rule for $\boldsymbol{v}_a$, to avoid difficulties in the derivation of the stability condition, we modify the notation of the weight update rule for $\boldsymbol{v}_a$ in (23) near the stable point $\boldsymbol{v}_a^*$ a little as follows:

$$v_i^{(t+1)} = v_i^{(t)} - \mu_i^{(t)} \varphi_i(u_i) c v_i^* v_i^{(t)}, \qquad 1 \le i \le M. \tag{31}$$

Here we assume that the learning rate $\mu_i^{(t)} (> 0)$ changes over time t and varies with different index i such that it satisfies $\mu_i^{(t)} v_i^{(t)} v_i^* = \mu_2$. The modification is justified because $v_i^{(t)} v_i^* \cong v_i^{*2}$ is positive when $v_i^{(t)}$ is near a stationary point v_i^*. Note that the modification applies only after $\boldsymbol{v}_a$ has reached sufficiently near a stable point $\boldsymbol{v}_a^*$.

Using the fact that $v_i^{(t)} = k_i^{(t)} v_i^*$ we can rewrite (31) as

$$k_i^{(t+1)} = [1 - \mu_i^{(t)} g_i(Z^{(t)}, \boldsymbol{k}^{(t)})] k_i^{(t)}, \qquad 1 \le i \le M \tag{32}$$

where

$$\begin{aligned} g_i(Z^{(t)}, \boldsymbol{k}^{(t)}) &= \varphi_i(u_i) c \\ &= \varphi_i((Z^{(t)} W^* \boldsymbol{x})_i + k_i^{(t)} v_i^* c) v_i^* c \end{aligned} \tag{33}$$

Using the weight update rules (29) and (32) for the new variables Z and K, the local stability condition is obtained in the following theorem.

Theorem 2: The local asymptotic stability of the stationary point of the proposed algorithm is governed by the nonlinear moment

$$\kappa_i = E\{\dot{\varphi}_i(e_i)\}E\{e_i^2\} - E\{\varphi_i(e_i)e_i\} \tag{34}$$

and it is stable if

$$1 + \kappa_i > 0, \quad 1 + \kappa_j > 0, \quad (1 + \kappa_i)(1 + \kappa_j) > 1 \tag{35}$$

for all $1 \leq i, j \leq N$. Thus the sufficient condition is

$$\kappa_i > 0, \quad 1 \leq i \leq N. \tag{36}$$

Proof: See Appendix II. ■

Because the condition for the stability of the ICA-FX in Theorem 2 is identical to that of the standard ICA in [30], the interpretation of the nonlinear moment κ_i can be consulted to [30]. Just stating the key point here, the local stability is preserved when the activation function $\varphi_i(e_i)$ is chosen to be positively correlated with the true activation function $\varphi_i^*(e_i) \triangleq -\dot{p}_i(e_i)/p_i(e_i)$.

Thus, as the standard ICA algorithm, the choice of activation function $\varphi_i(e_i)$ is of great importance, and the performance of ICA-FX depends heavily on the function $\boldsymbol{\varphi}(\boldsymbol{e})$, which is determined by the densities $p_i(e_i)$'s. But in practical situations, these densities are mostly unknown, and true densities are approximated by some model densities, generally given by (i) momentum expansion, (ii) a simple parametric model not far from Gaussian, or (iii) a mixture of simple parametric models [31]. In this work, we do not need an exact approximation of the density $p_i(u_i)$ because we do not have physical sources like in BSS problems. Therefore, we use the extended Infomax algorithm [26], one of the approximation methods belonging to type (ii), because of its computational efficiency and wide applications.

Now, we discuss the properties of the ICA-FX in terms of the suitability of the proposed algorithm for the classification problems.

D. Properties of ICA-FX

In ICA-FX, given a new instance consisting of N features $\boldsymbol{x} = [x_1, \cdots, x_N]$, we transform it into an M-dimensional new feature vector $\boldsymbol{f}_a = [f_1, \cdots, f_M]$ and use it to estimate which class the instance belongs to. In the following, we discuss why ICA-FX is suitable for the classification problems in the statistical sense by showing that the new feature f_i contains information about class label c under sub- or super-Gaussian density of u_i.

Consider a normalized zero-mean binary output class c, with its density

$$p_c(c) = p_1\delta(c - c_1) + p_2\delta(c - c_2), \tag{37}$$

where $\delta(\cdot)$ is a dirac delta function, and p_1, p_2 are the probabilities that class c takes values c_1 and c_2, respectively.

Suppose that u_i $(i = 1, \cdots, N)$ has density $p_i(u_i)$, which is sub-Gaussian ($p_i(u_i) \propto N(\mu, \sigma^2) + N(-\mu, \sigma^2)$) or super-Gaussian ($p_i(u_i) \propto N(0, \sigma^2)\mathrm{sech}^2(u_i)$) as in [26], where $N(\mu, \sigma^2)$ is the normal density with mean μ and variance σ^2. Then the density of f_i $(i = 1, \cdots, M)$ is proportional to the convolution of two densities $p_i(u_i)$ and $p_c(-c/w_{i,N+1})$ by the assumption that u_i's and c are independent; i.e.,

$$\begin{aligned} p(f_i) =& \frac{1}{|w_{i,N+1}|} p_i(u_i) * p_c(-\frac{c}{w_{i,N+1}}) \\ \propto& \begin{cases} p_1 N(-w_{i,c}c_1, \sigma^2)\mathrm{sech}^2(f_i + w_{i,N+1}c_1) & \\ \quad + p_2 N(-w_{i,N+1}c_2, \sigma^2)\mathrm{sech}^2(f_i + w_{i,N+1}c_2) & \text{if } p_i(u_i)\text{: super-Gaussian} \\ p_1 N(\mu - w_{i,N+1}c_1, \sigma^2) + p_2 N(\mu - w_{i,N+1}c_2, \sigma^2) & \\ \quad + p_1 N(-\mu - w_{i,N+1}c_1, \sigma^2) + p_2 N(-\mu - w_{i,N+1}c_2, \sigma^2) & \text{if } p_i(u_i)\text{: sub-Gaussian} \end{cases} \end{aligned} \tag{38}$$

because $f_i = u_i - w_{i,N+1}c$.

Figure 3 shows the densities of super- and sub-Gaussian models of u_i and the corresponding densities of f_i for varying $w_{i,N+1} = [0 \cdots 4]$. In the figure, we set $\mu = 1$, $\sigma = 1$, $p_1 = p_2 = 0.5$, and $c_1 = -c_2 = 1$. We can see in Fig. 3 that super-Gaussian is sharper than sub-Gaussian at peak. For the super-Gaussian model of u_i, we can see that as $w_{i,N+1}$ grows, the density of f_i has two peaks, which are separated from each other, and the shape

is quite like a sub-Gaussian model with a large mean. For the sub-Gaussian model of u_i, we can see that it also takes two peaks as the weight $w_{i,N+1}$ grows, though the peaks are smoother than those of super-Gaussian. In both cases, as $w_{i,N+1}$ grows, the influence of output class c becomes dominant in the density of f_i, and the classification problem becomes easier: for a given f_i check if it is larger than zero and then associate it with the corresponding class c.

This phenomenon can be interpreted as a discrete source estimation problem in a noisy channel, as shown in Fig. 4. If we regard class c as an input and u_i as noise, our goal is to estimate c through channel output f_i. Because we assumed that c and u_i's are independent, the higher the signal-to-noise ratio (SNR) becomes, the more class information is conveyed in the channel output f_i. The SNR can be estimated using powers of source and noise, which in this case leads to the following estimation:

$$SNR = \frac{E\{c^2\}}{E\{(u_i/w_{i,N+1})^2\}}. \tag{39}$$

Therefore, if we can make large $w_{i,N+1}$, the noise power in Fig. 4 is suppressed and we can easily estimate the source c.

In many real-world problems, as the number of input features increases, the contribution of class c to u_i becomes small; i.e., $w_{i,N+1}$ becomes relatively small such that the density of f_i is no longer bimodal. Even if this is the case, the density has a flatter top that looks like a sub-Gaussian density model, which is easier to estimate classes than those with normal densities.

IV. Experimental Results

In this section we will present some experimental results which show the characteristics of the proposed algorithm. In order to show the effectiveness of the proposed algorithm, we selected the same number of features from both the original features and the extracted features and compared the classification performances. In the selection of features for original data, we used the MIFS-U (mutual information feature selector under uniform information distribution) [32], [33] which makes use of the mutual information between input features and output class in ordering the significance of features. It is noted that the simulation results can vary depending on the initial condition of the rate updating

rule because there may be many local optimum solutions.

A. Simple problem

Suppose we have two input features x_1 and x_2 uniformly distributed on [-1,1] for a binary classification, and the output class y is determined as follows:

$$y = \begin{cases} 0 & \text{if } x_1 + x_2 < 0 \\ 1 & \text{if } x_1 + x_2 \geq 0. \end{cases}$$

Here, $y = 0$ corresponds to $c = -1$ and $y = 1$ corresponds to $c = 1$.

Plotting this problem on a three-dimensional space of (x_1, x_2, y) leads to Fig. 5 where the class information, as well as the input features, correspond to each axis, respectively. The data points are located in the shaded areas in this problem. As can be seen in the figure, this problem is linearly separable, and we can easily distinguish $x_1 + x_2$ as an important feature. But feature extraction algorithms based on conventional unsupervised learning, such as the conventional PCA and ICA, cannot extract $x_1 + x_2$ as a new feature because they only consider the input distribution; i.e., they only examine (x_1, x_2) space. For problems of this kind, feature selection methods in [32], [33] also fail to find adequate features because they have no ability to construct new features by themselves. Note that other feature extraction methods using supervised algorithms such as LDA and MMI can solve this problem.

For this problem, we performed ICA-FX with $M = 1$ and could get $u_1 = 43.59x_1 + 46.12x_2 + 36.78y$ from which a new feature $f_1 = 43.59x_1 + 46.12x_2$ is obtained. To illustrate the characteristic of ICA-FX on this problem, we plotted u_1 as a thick arrow in Fig. 5 and f_1 is the projection of u_1 onto the (x_1, x_2) feature space.

B. IBM datasets

These datasets were generated by Agrawal et al. [34] to test their data mining algorithm CDP . Each of the datasets has nine attributes: *salary, commission, age, education level, make of the car, zipcode of the town, value of the house, years house owned,* and *total amount of the loan.* We have downloaded the data generation code from [35] and tested the proposed algorithm for several datasets generated by the code. The datasets used in

our experiments are shown in Table I.

As can be seen from Table I, these datasets are linearly separable and use only a few features for classification. We generated 1000 instances for each dataset with noise of zero mean and either 0% or 10% of SNR added to the attributes, among which 66% were used as training data while the others were reserved for test. In the training, we used C4.5 [36], one of the most popular decision-tree algorithms which gives deterministic classification rules, and a three-layered MLP. To show the effectiveness of our feature extraction algorithm, we have compared the performance of ICA-FX with PCA, LDA, and the original data with various number of features. For the original data, we applied the feature selection algorithm MIFS-U, which selects good features among candidate features, before training. In training C4.5, all the parameters were set as the default values in [36], and for MLP, three hidden nodes were used with a standard back-propagation (BP) algorithm with zero momentum and a learning rate of 0.2. After 300 iterations, we stopped training the network.

The experimental results are shown in Table II. In the table, we compared the performance of the original features selected with MIFS-U and the newly extracted features with PCA, LDA, and ICA-FX. Because this is a binary classification problem, standard LDA extracts only one feature for all cases. The classification performances on the test set trained with C4.5 and BP are presented in Table II. The parentheses after the classification performance of C4.5 contain the size of the decision tree.

As can be seen from Table II, C4.5 and BP produce similar classification performances on these data sets. For all three of the problems, ICA-FX outperformed other methods. We also can see that PCA performed worst in all cases, even worse than the original features selected with MIFS-U. This is because PCA can be thought as a result of unsupervised learning, and the ordering of its principle components has nothing to do with the classification. Note that the performances with 'all' features are different for different feature extraction/selection methods, although they operate on the same space of all the features. They operate on the same amount of information about the class. But the classifier systems do not make full use of the information.

In the cases of 0% noise power, with only one feature we achieved very good performance

for all the cases. In fact, in IBM1 and IBM2, the first feature selected among the original ones was *salary*, while the newly extracted feature with $M = 1$ corresponds to ($salary + commission$) and ($salary + commission - 6500 \times ed\_level$), respectively. Comparing these with Table I, we can see these are very good features for classification. The small numbers of tree size for extracted features compared to that for the other methods show our feature extraction algorithm can be utilized to generate oblique decision trees resulting in rules easy to understand. For the case of 10% SNR, ICA-FX also performed better than others in most cases. From these results, we can see that ICA-FX performs excellently, especially for linearly separable problems.

C. UCI datasets

The UCI machine learning repository contains many real-world data sets that have been used by numerous researchers [37]. In this subsection, we present experimental results of the proposed extraction algorithm for some of these data sets. Table III shows the brief information of the data sets used in this paper. We conducted conventional PCA, ICA, and LDA algorithms on these datasets and extracted various numbers of features and compared the classification performances with that of the ICA-FX. Because there is no measure on relative importance among independent components from ICA, we used MIFS-U in selecting the important features for the classification. For comparison, we have also conducted MIFS-U on the original datasets and report the performance.

As classifier systems, we used MLP, C4.5, and SVM. For all the classifiers, input values of the data were normalized to have zero means and standard deviations of one. In training MLP, the standard BP algorithm was used with three hidden nodes, two output nodes, a learning rate of 0.05, and a momentum of 0.95. We trained the networks for 1,000 iterations. The parameters of C4.5 were set to default values in [36]. For SVM, we used 'mySVM' program by Stefan Ruping of University of Dortmund [38]. For the kernel function we used radial (Gaussian) kernel and the other parameters were set as default. Because the performance of the radial kernel SVM critically depends on the value of γ, we have conducted SVM with various values of $\gamma = 0.01 \sim 1$ and report the maximum classification rate. Thirteen-fold cross-validation was used for the sonar dataset and ten-fold cross-validation was used for the others. For MLP, ten experiments were conducted

for each dataset and the averages and the standard deviations are reported in this paper.

C.1 Sonar Target data

The sonar target classification problem is described in [39]. This data set was constructed to discriminate between the sonar returns bounced off a metal cylinder and those bounced off a rock. It consists of 208 instances, with 60 features and two output classes: *mine/rock*. In our experiment, we used 13-fold cross validation in getting the performances as follows. The 208 instances were divided randomly into 13 disjoint sets with 16 cases in each. For each experiment, 12 of these sets are used as training data, while the 13th is reserved for testing. The experiment is repeated 13 times so that every case appears once as part of a test set.

The training was conducted with MLP, C4.5, and SVM for various numbers of features. Table IV shows the result of our experiment. The reported performance for MLP is an average over the 10 experiments and the numbers in parentheses denote the standard deviation. The result shows that the extracted features from ICA-FX perform better than the original ones, especially when the number of features to be selected is small. In the table, we can see that the performances of ICA-FX are almost the same for small numbers of features and far better than when all the 60 features were used. From this phenomenon, we can infer that all the available information about the class is contained in the first feature.

Note that the performances of unsupervised feature extraction methods PCA and ICA are not as good as expected. From this, we can see that the unsupervised methods of feature extraction are not good choices for the classification problems.

The first three figures in Fig. 6 are the estimates of conditional densities $p(f|c)$'s (class-specific density estimates) of the first selected feature among the original features by MIFS-U (which is the 11th of 60 features), the feature extracted by LDA, and the feature extracted by ICA-FX with $M = 1$. We conducted the density estimates with the well known Parzen window method [40] using both training and test data. In applying Parzen window, the window width parameter was set to 0.2. The result shows that the conditional density of the feature from ICA-FX is much more balanced than those of the original and LDA in the feature space. In the figures of 6.(a),(b),(c), if the domain for $p(f|c = 0) \neq$

0 and the domain for $p(f|c = 1) \neq 0$ do not overlap, then we can make no error in classification. We can see that the overlapping region of the two classes is much smaller in ICA-FX than the other two. This is why the performance of ICA-FX is far better than the others with only one feature. We also present the density estimate $p(f)$ of the feature from ICA-FX in Fig. 6(d). Note that in Fig. 6(d), the distribution of the feature from ICA-FX is much flatter than the Gaussian distribution and looks quite like the density of feature f_i obtained with sub-Gaussian model. The dotted line of Fig. 6(d) is the density of sub-Gaussian model shown in Fig. 3(d) with $w_{i,N+1} = 1.5$.

C.2 Wisconsin Breast Cancer data

This database was obtained from the University of Wisconsin Hospitals, Madison, from Dr. William H. Wolberg [41]. The data set consists of nine numerical attributes and two classes, which are benign and malignant. It contains 699 instances with 458 benign and 241 malignant. There are 16 missing values in our experiment and we replaced these with average values of corresponding attributes.

We compared the performances of ICA-FX with those of PCA, ICA, LDA, and the original features selected with MIFS-U. The classification results are shown in Table V. As in the sonar dataset, we trained the data with C4.5, MLP, and SVM. The meta-parameters for C4.5, MLP, and SVM are the same as those for the sonar problem. For verification, 10-fold cross validation is used. In the table, classification performances are present and the numbers in parentheses are standard deviations of MLP over 10 experiments.

The result shows that with only one extracted feature, we can get nearly the maximum classification performance that can be achieved with at least two or three original features. The performance of LDA is almost the same as ICA-FX for this problem.

C.3 Pima Indian Diabetes data

This data set consists of 768 instances in which 500 are class 0 and the other 268 are class 1. It has 8 numeric features with no missing value.

For this data, we applied PCA, ICA, LDA, and ICA-FX, and compared their performances. Original features selected by MIFS-U were also compared. In training, we used C4.5, MLP, and SVM. The meta-parameters for the classifiers were set to be equal to the

previous cases. For verification, 10-fold cross validation was used.

In Table VI, classification performances are presented. As shown in the table, the performance of ICA-FX is better than those of other methods regardless of what classifier system was used when the number of features is small. We can also see that the performances of different methods get closer as the number of extracted features becomes large. Note also that for ICA-FX, the classification rate of one feature is as good as those of the other cases where more features are used.

V. Conclusions

In this paper, we have proposed an algorithm ICA-FX for feature extraction and have presented the stability condition for the proposed algorithm. The proposed algorithm is based on the standard ICA and can generate very useful features for classification problems.

Although ICA can be directly used for feature extraction, it does not generate useful information because of its unsupervised learning nature. In the proposed algorithm, we added class information in training ICA. The added class information plays a critical role in the extraction of useful features for classification. With the additional class information we can extract new features containing maximal information about the class. The number of extracted features can be arbitrarily chosen.

The stability condition for the proposed algorithm suggests that the activation function $\varphi_i(\cdot)$ should be chosen to well represent the true density of the source. If we are to use a squashing function such as sigmoid or logistic as an activation function, the true source density should not be Gaussian. If it is so, the algorithm diverges as in standard ICA.

Since it uses the standard feed-forward structure and learning algorithm of ICA, it is easy to implement and train. Experimental results for several data sets show that the proposed algorithm generates good features that outperform the original features and other features extracted from other methods for classification problems. Because the original ICA is ideally suited for processing large datasets such as biomedical ones, the proposed algorithm is also expected to perform well for large-scale classification problems.

The proposed algorithm has been developed for two-class problems, and more work is needed to extend the proposed method for multiclass problems. One possible approach may start from appropriately choosing a coding scheme for multiclass labels.

References

[1] V.S. Cherkassky and I.F. Mulier, *Learning from Data*, chapter 5, John Wiley & Sons, 1998.

[2] G.H. John, *Enhancements to the data mining process*, Ph.D. thesis, Computer Science Dept., Stanford University, 1997.

[3] I.T. Joliffe, *Principal Component Analysis*, Springer-Verlag, 1986.

[4] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, second edition, 1990.

[5] H. Lu, R. Setiono, and H. Liu, "Effective data mining using neural networks," *IEEE Trans. Know. and Data Eng.*, vol. 8, no. 6, Dec. 1996.

[6] J.M. Steppe, K.W. Bauer Jr., and S.K. Rogers, "Integrated feature and architecture selection," *IEEE Trans. Neural Networks*, vol. 7, no. 4, July 1996.

[7] K.J. McGarry, S. Wermter, and J. MacIntyre, "Knowledge extraction from radial basis function networks and multi-layer perceptrons," in *Proc. Int'l Joint Conf. on Neural Networks 1999*, Washington D.C., July 1999.

[8] R. Setiono and H. Liu, "A connectionist approach to generating oblique decision trees," *IEEE Trans. Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 29, no. 3, June 1999.

[9] Q. Li and D.W. Tufts, "Principal feature classification," *IEEE Trans. Neural Networks*, vol. 8, no. 1, Jan. 1997.

[10] M. Baldoni, C. Baroglio, D. Cavagnino, and L. Saitta, *Towards automatic fractal feature extraction for image recognition*, pp. 357 – 373, Kluwer Academic Publishers, 1998.

[11] Y. Mallet, D. Coomans, J. Kautsky, and O. De Vel, "Classification using adaptive wavelets for feature extraction," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 10, Oct. 1997.

[12] A.J. Bell and T.J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, June 1995.

[13] A. Hyvarinen, E. Oja, P. Hoyer, and J. Hurri, "Image feature extraction by sparse coding and independent component analysis," in *Proc. Fourteenth International Conference on Pattern Recognition*, Brisbane, Australia, Aug. 1998.

[14] M. Kotani et. al, "Application of independent component analysis to feature extraction of speech," in *Proc. Int'l Joint Conf. on Neural Networks 1999*, Washington D.C., July 1999.

[15] A.D. Back and T.P. Trappenberg, "Input variable selection using independent component analysis," in *Proc. Int'l Joint Conf. on Neural Networks 1999*, Washington D.C., July 1999.

[16] H.H. Yang and J. Moody, "Data visualization and feature selection: new algorithms for nongaussian data," *Advances in Neural Information Processing Systems*, vol. 12, 2000.

[17] J.W. Fisher III and J.C. Principe, "A methodology for information theoretic feature extraction," in *Proc. Int'l Joint Conf. on Neural Networks 1998*, Anchorage, Alasca, May 1998.

[18] K. Torkkola and W.M. Campbell, "Mutual information in learning feature transformations," in *Proc. Int'l Conf. Machine Learning*, Stanford, CA, 2000.

[19] N. Kwak, C.-H. Choi, and C.-Y. Choi, "Feature extraction using ica," in *Proc. Int'l Conf. on Artificial Neural Networks 2001*, Vienna Austria, Aug. 2001.

[20] J. Herault and C. Jutten, "Space or time adaptive signal provessing by neural network models," in *Proc. AIP Conf. Neural Networks Computing*, Snowbird, UT, USA, 1986, vol. 151, pp. 206–211.

[21] J. Cardoso, "Source separation using higher order moments," in *Proc. ICASSP*, 1989, pp. 2109–2112.

[22] P. Comon, "Independent component analysis, a new concept?," *Signal Processing*, vol. 36, pp. 287–314, 1994.

[23] D. Obradovic and G. Deco, "Blind source seperation: are information maximization and redundancy mini-

mization different?," in *Proc. IEEE Workshop on Neural Networks for Signal Processing 1997*, Florida, Sept. 1997.

[24] J. Cardoso, "Infomax and maximum likelifood for blind source separation," *IEEE Signal Processing Letters*, vol. 4, no. 4, April 1997.

[25] T.-W. Lee, M. Girolami, A.J. Bell, and T.J. Sejnowski, "A unifying information -theretic framework for independent component analysis," *Computers and Mathematics with Applications*, vol. 31, no. 11, March 2000.

[26] T-W. Lee, M. Girolami, and T.J. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources," *Neural Computation*, vol. 11, no. 2, Feb. 1999.

[27] M. Girolami, "An alternative perspective on adaptive independent component analysis algorithms," *Neural Computation*, vol. 10, no. 8, pp. 2103 –2114, 1998.

[28] L. Xu, C. Cheung, and S.-I. Amari, "Learned parametric mixture based ica algorithm," *Neurocomputing*, vol. 22, no. 1-3, pp. 69 – 80, 1998.

[29] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991.

[30] J. Cardoso, "On the stability of source separation algorithms," *Journal of VLSI Signal Processing Systems*, vol. 26, no. 1, pp. 7 – 14, Aug. 2000.

[31] N. Vlassis and Y. Motomura, "Efficient source adaptivity in independent component analysis," *IEEE Trans. Neural Networks*, vol. 12, no. 3, May 2001.

[32] N. Kwak and C.-H. Choi, "Improved mutual information feature selector for neural networks in supervised learning," in *Proc. Int'l Joint Conf. on Neural Networks 1999*, Washington D.C., July 1999.

[33] N. Kwak and C.-H. Choi, "Input feature selection for classification problems," *IEEE Trans. Neural Networks*, vol. 13, no. 1, Jan. 2002.

[34] R. Agrawal, T. Imielinski, and A. Swami, "Database mining: a performance perspective," *IEEE Trans. Know. and Data Eng.*, vol. 5, no. 6, Dec. 1993.

[35] Quest Group at IBM Almaden Research Center, "Quest synthetic data generation code for classification," 1993, For information contact http://www.almaden.ibm.com/cs/quest/.

[36] R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

[37] P. M. Murphy and D. W. Aha, "Uci repository of machine learning databases," 1994, For more information contact ml-repository@ics.uci.edu or http://www.cs.toronto.edu/∼delve/.

[38] Stefan Ruping, "mysvm – a support vector machine," For more information contact http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/.

[39] R.P. Gorman and T.J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural Networks*, vol. 1, pp. 75–89, 1988.

[40] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statistics*, vol. 33, pp. 1065–1076, Sept. 1962.

[41] W.H. Wolberg and O.L.Mangasarian, "Multisurface method of pattern separation for medical diagnosis applied to breast cytology," *Proc. National Academy of Sciences*, vol. 87, Dec. 1990.

Appendix

I. Proof of Theorem 1

If (27) is to be a stationary point of learning rule (23), $\Delta W \triangleq W^{(t+1)} - W^{(t)}$ and $\Delta \boldsymbol{v} \triangleq \boldsymbol{v}^{(t+1)} - \boldsymbol{v}^{(t)}$ must be zero in the statistical sense. Thus

$$\begin{aligned} E\{[I_N - \boldsymbol{\varphi}(\boldsymbol{u})\boldsymbol{f}^T]W\} &= 0 \\ E\{\boldsymbol{\varphi}(\boldsymbol{u}_a)c\} &= 0 \end{aligned} \tag{40}$$

must be satisfied. The second equality is readily satisfied because of the independence of $\boldsymbol{u}_a$ and c and the zero mean assumption on c. The first equality holds if

$$E\{I_N - \boldsymbol{\varphi}(\boldsymbol{u})\boldsymbol{f}^T\} = I_N - E\{\boldsymbol{\varphi}(\boldsymbol{u})\boldsymbol{u}^T\} - E\{\boldsymbol{\varphi}(\boldsymbol{u})c\}\boldsymbol{v}^T = 0. \tag{41}$$

In the equation the last term $E\{\boldsymbol{\varphi}(\boldsymbol{u})c\} = 0$ because $\boldsymbol{u}$ and c are independent and c is a zero mean random variable. Thus, the condition (41) holds if

$$E\{\varphi_i(u_i)u_j\} = \delta_{ij}, \tag{42}$$

where δ_{ij} is a Kronecker delta. When $i \neq j$, this condition is satisfied because of the independence assumption on $u_i (= e_i)$'s, and the remaining condition is

$$E\{\varphi_i(u_i)u_i\} = E\{\varphi_i(\lambda_i s_{\Pi(i)})\lambda_i s_{\Pi(i)}\} = 1, \qquad \forall 1 \leq i \leq N. \tag{43}$$

Here we used the fact that $u_i = e_i = \lambda_i s_{\Pi(i)}$, where λ_i is the ith diagonal element of scaling matrix Λ and $s_{\Pi(i)}$ is the ith signal permuted through Π.

Assuming that s_i has an even pdf, then u_i has an even pdf and $\varphi_i (= \dot{p}_i(u_i)/p_i(u_i))$ is an odd function. Therefore, λ_i that satisfies (43) always comes in pairs: if λ is a solution, so is $-\lambda$. Furthermore if we assume that φ_i is an increasing differentiable function, (43) has a unique solution λ_i^* up to a sign change.

II. Proof of Theorem 2

For the proof, we use a standard tool for analyzing the local asymptotic stability of a stochastic algorithm. It makes use of the derivative of the mean field at a stationary point. In our problem, $Z \in \Re^{N \times N}$ and $\boldsymbol{k} \in \Re^M$ constitute an $N \times N + M$ dimensional space, and we

can denote this space as a direct sum of Z and $\boldsymbol{k}$; i.e., $Z \oplus \boldsymbol{k}$. Then the derivative considered here is that of a mapping $H : Z \oplus \boldsymbol{k} \to E\{G(Z,\boldsymbol{k})Z\} \oplus E\{g_1(Z,\boldsymbol{k})k_1\} \oplus \cdots \oplus E\{g_M(Z,\boldsymbol{k})k_M\}$ at the stationary point $(Z^*, \boldsymbol{k}^*)$ where $Z^* = I_N$ and $\boldsymbol{k}^* = \mathbf{1}_M = [1, \cdots, 1]^T$. The derivative is of $(N \times N + M)^2$ dimension, and if it is positive definite, the stationary point is a local asymptotic stable point. As written in [30], because the derivative of the mapping H is very sparse, we can use the first-order expansion of H at the point $(Z^*, \boldsymbol{k}^*)$ rather than trying to use the exact derivatives.

For convenience, let us split H into two functions H^1 and H^2 such that

$$\begin{aligned} H^1 &: Z \oplus \boldsymbol{k} \to E\{G(Z,\boldsymbol{k})Z\} \in \Re^{N \times N} \\ H_i^2 &: Z \oplus \boldsymbol{k} \to E\{g_i(Z,\boldsymbol{k})k_i\}, \qquad 1 \le i \le M. \end{aligned} \tag{44}$$

Note that $H = H^1 \oplus H^2$. To get the first order linear approximation of the function at a stationary point $(Z^*, \boldsymbol{k}^*)$, we evaluate H^1 and H^2 near a small variation of the stationary point $(Z, \boldsymbol{k}) = (Z^* + \mathcal{E}, \boldsymbol{k}^* + \boldsymbol{\varepsilon})$, where $\mathcal{E} \in \Re^{N \times N}$ and $\boldsymbol{\varepsilon} \in \Re^M$.

$$\begin{aligned} & H_{ij}^1(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon}) \\ &\quad = [E\{G(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon})\}(I_N + \mathcal{E})]_{ij} \\ &\quad = [E\{G(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon})\}]_{ij} + [E\{G(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon})\}\mathcal{E}]_{ij} \\ &\quad = E\{G_{ij}\} + \sum_{n=1}^{N}\sum_{m=1}^{N} E\{\frac{\partial G_{ij}}{\partial Z_{nm}}\}\mathcal{E}_{nm} + \sum_{m=1}^{M} E\{\frac{\partial G_{ij}}{\partial k_m}\}\varepsilon_m + \sum_{m=1}^{N} E\{G_{im}\}\mathcal{E}_{mj} \\ &\qquad + o(\mathcal{E}) + o(\boldsymbol{\varepsilon}). \end{aligned} \tag{45}$$

and

$$\begin{aligned} & H_i^2(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon}) \\ &\quad = E\{g_i(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon})\}(1 + \varepsilon_i) \\ &\quad = E\{g_i(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon})\} + E\{g_i(I_N + \mathcal{E}, \mathbf{1}_M + \boldsymbol{\varepsilon})\}\varepsilon_i \\ &\quad = E\{g_i\} + \sum_{n=1}^{N}\sum_{m=1}^{N} E\{\frac{\partial g_i}{\partial Z_{mn}}\}\mathcal{E}_{mn} + \sum_{m=1}^{M} E\{\frac{\partial g_i}{\partial k_m}\}\varepsilon_m + E\{g_i\}\varepsilon_i + o(\mathcal{E}) + o(\boldsymbol{\varepsilon}). \end{aligned} \tag{46}$$

Using the independence and zero mean assumptions on e_i's and c, these can be further

expanded as

$$
\begin{aligned}
&H^1_{ij}(I_N+\mathcal{E},\mathbf{1}_M+\boldsymbol{\varepsilon}) \\
&= \begin{cases}
\mathcal{E}_{ij}E\{\dot\varphi_i(e_i)e_j^2\}+E\{\varphi_i(e_i)e_i\}\mathcal{E}_{ji}+v_j^*\sum_{m=1}^M\mathcal{E}_{im}v_m^*E\{\dot\varphi_i(e_i)c^2\} \\
\qquad -\varepsilon_i v_i^* v_j^* E\{\dot\varphi_i(e_i)c^2\}+o(\mathcal{E})+o(\boldsymbol{\varepsilon}) & \text{if} \quad 1\le i,j\le M \\
\mathcal{E}_{ij}E\{\dot\varphi_i(e_i)e_j^2\}+E\{\varphi_i(e_i)e_i\}\mathcal{E}_{ji}+v_j^*\sum_{m=1}^M\mathcal{E}_{im}v_m^*E\{\dot\varphi_i(e_i)c^2\} \\
\qquad +o(\mathcal{E})+o(\boldsymbol{\varepsilon}) & \text{if} \quad M<i\le N,\ 1\le j\le M \\
\mathcal{E}_{ij}E\{\dot\varphi_i(e_i)e_j^2\}+E\{\varphi_i(e_i)e_i\}\mathcal{E}_{ji}+o(\mathcal{E})+o(\boldsymbol{\varepsilon}) & \text{if} \quad M<i,j\le N
\end{cases}
\end{aligned} \tag{47}
$$

and

$$
\begin{aligned}
&H^2_i(I_N+\mathcal{E},\mathbf{1}_M+\boldsymbol{\varepsilon}) \\
&= -v_i^*\sum_{m=1}^M\mathcal{E}_{im}v_m^*E\{\dot\varphi_i(e_i)c^2\}+\varepsilon_i v_i^{*2}E\{\dot\varphi_i(e_i)c^2\}+o(\mathcal{E})+o(\boldsymbol{\varepsilon}) \qquad 1\le i\le M.
\end{aligned} \tag{48}
$$

Now, we develop the local stability conditions case by case.

(Case 1) $i,j>M$

In this case, H^1_{ij} and H^1_{ji} only depend on $\mathcal{E}_{ij}$ and $\mathcal{E}_{ji}$ and are represented as

$$
\begin{aligned}
&\begin{bmatrix}H^1_{ij}\\ H^1_{ji}\end{bmatrix}=\begin{bmatrix}E\{\dot\varphi_i(e_i)\}E\{e_j^2\} & E\{\varphi_i(e_i)e_i\}\\ E\{\varphi_i(e_j)e_j\} & E\{\dot\varphi_j(e_j)\}E\{e_i^2\}\end{bmatrix}\begin{bmatrix}\mathcal{E}_{ij}\\ \mathcal{E}_{ji}\end{bmatrix}\triangleq D_{ij}\begin{bmatrix}\mathcal{E}_{ij}\\ \mathcal{E}_{ji}\end{bmatrix} \qquad \text{if} \quad i\neq j \\
&H^1_{ii}=[E\{\dot\varphi_i(e_i)e_i^2\}+E\{\varphi_i(e_i)e_i\}]\mathcal{E}_{ii}\triangleq d_i\mathcal{E}_{ii}.
\end{aligned} \tag{49}
$$

Thus for $i\neq j$, Z_{ij} and Z_{ji} are stabilized when D_{ij} is positive definite. And if $i=j$, Z_{ii} is stabilized when d_i is positive. Using the fact that $E\{\varphi_i(e_i)e_i\}=1\ \forall i=1,\cdots,N$, we can show that the local stability condition for the pair (i,j) when $i,j>M$ is (35).

(Case 2) $i\le M, j>M$

In this case, H^1_{ij} and H^1_{ji} are dependent not only on $\mathcal{E}_{ij}$ and $\mathcal{E}_{ji}$ but also on all $\mathcal{E}_{jm}$, $m=1,\cdots,M$. Thus for a fixed j, we augment all the H^1_{ij} and H^1_{ji}, $i=1,\cdots,M$, and construct a $2M$-dimensional vector $\boldsymbol{H}_j\triangleq[H^1_{1j},\cdots,H^1_{Mj},H^1_{j1},\cdots,H^1_{jM}]^T$. Now this augmented vector $\boldsymbol{H}_j$ depends only on $\boldsymbol{\mathcal{E}}_j\triangleq[\mathcal{E}^1_{1j},\cdots,\mathcal{E}_{Mj},\mathcal{E}_{j1},\cdots,\mathcal{E}_{jM}]^T$ and can be represented as a linear equation $\boldsymbol{H}_j=\boldsymbol{D}_j\boldsymbol{\mathcal{E}}_j$, using an appropriate matrix $\boldsymbol{D}_j\in\Re^{2M\times 2M}$. The stability of $\boldsymbol{Z}_j=[Z_{1j},\cdots,Z_{Mj},Z_{j1},\cdots,Z_{jM}]^T$ for $j>M$ is equivalent to the positive definiteness of $\boldsymbol{D}_j$ and it can be checked by investigating the sign of the $\boldsymbol{H}_j^T\boldsymbol{\mathcal{E}}_j$.

Substituting (47) and using $E\{\varphi_i(e_i)e_i\} = 1\ \forall i = 1, \cdots, N$, we get

$$\begin{aligned} \boldsymbol{H}_j^T \boldsymbol{\mathcal{E}}_j &= \sum_{i=1}^{M} (H_{ij}^1 \mathcal{E}_{ij} + H_{ji}^1 \mathcal{E}_{ji}) \\ &= \sum_{i=1}^{M} [E\{\dot{\varphi}_i(e_i)e_j^2\}\mathcal{E}_{ij}^2 + 2\mathcal{E}_{ij}\mathcal{E}_{ji} + E\{\dot{\varphi}_j(e_j)e_i^2\}\mathcal{E}_{ji}^2] + E\{\dot{\varphi}_j(e_j)\}E\{c^2\}(\sum_{i=1}^{M} \mathcal{E}_{ji} v_i^*)^2. \end{aligned} \tag{50}$$

If we assume that $\dot{\varphi}_j(\cdot)$ is nonnegative, as we did in the proof of the uniqueness of the scalar λ_j, the last term is nonnegative. Thus, a sufficient condition for this equation to be positive is to make the first term positive, and this condition is satisfied if and only if equation (35) holds. Therefore, (35) becomes a sufficient condition for the local stability of $\boldsymbol{Z}_j$.

(Case 3) $i, j \leq M$

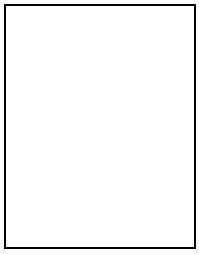
In this case, because H_{ij}^1 and H_i^2 are dependent both on $\mathcal{E}$ and $\boldsymbol{\varepsilon}$, we construct a new vector and investigate the stability condition of the vector as in the previous case.

Consider the $M \times M + M$ dimensional vectors $\boldsymbol{H} \triangleq [H_{11}^1, H_{12}^1, \cdots, H_{MM}^1, H_1^2, \cdots, H_M^2]^T$ and $\boldsymbol{\mathcal{E}} \triangleq [\mathcal{E}_{11}, \mathcal{E}_{12}, \cdots, \mathcal{E}_{MM}, \varepsilon_1, \cdots, \varepsilon_M]^T$. Using (47) and (48), $\boldsymbol{H}$ can be represented as the linear equation $\boldsymbol{H} = \boldsymbol{D}\boldsymbol{\mathcal{E}}$, where $\boldsymbol{D}$ is an appropriate matrix. Thus, the stability of the $\mathcal{Z} = [Z_{11}, Z_{12} \cdots, Z_{MM}]^T$ and $\boldsymbol{k}$ can be checked using the same procedure as the previous case.

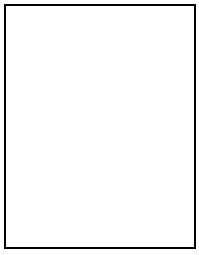
$$\begin{aligned} \boldsymbol{H}^T \boldsymbol{\mathcal{E}} &= \sum_{i=1}^{M}\sum_{j=1}^{M} H_{ij}^1 \mathcal{E}_{ij} + \sum_{i=1}^{M} H_i^2 \varepsilon_i \\ &= \sum_{i=1}^{M}\sum_{j=1}^{M} (\mathcal{E}_{ij}^2 E\{\dot{\varphi}_i(e_i)e_j^2\} + \mathcal{E}_{ij}\mathcal{E}_{ji}) + \sum_{i=1}^{M} [E\{\dot{\varphi}_i(e_i)\}E\{c^2\}(v_i^*\varepsilon_i - \sum_{j=1}^{M} v_j^* \mathcal{E}_{ij})^2] \end{aligned} \tag{51}$$

The last term is nonnegative with the assumption of $\dot{\varphi}_i(\cdot) \geq 0$, and a sufficient condition for the double summation to be positive is (35). Thus, $\mathcal{Z} \oplus \boldsymbol{k}$ is locally stable if condition (35) holds.

Combining the stability conditions for the case 1, 2, and 3, we conclude that the learning rule (23) for ICA-FX is locally asymptotically stable at the stationary point (27) if condition (35) holds.



Nojun Kwak received the B.S. and M.S. degrees from the School of Electrical Engineering and Computer Science, Seoul National University, Seoul, Korea, in 1997 and 1999 respectively. He is currently pursuing the Ph.D. degree at Seoul National University. His research interests include neural networks, machine learning, data mining, image processing, and their applications. IEEE Membership Number : Student 41285556



Chong-Ho Choi received the B.S. degree from Seoul National University, Seoul, Korea, in 1970 and the M.S. and Ph.D. degrees from the University of Florida, Gainesville, in 1975 and 1978 respectively. He was a Senior Researcher with the Korea Institute of Technology from 1978 to 1980. He is currently a Professor in the School of Electrical Engineering and Computer Science, Seoul National University. He is also affiliated with the Automation and Systems Research Institute, Seoul National University. His research interests include control theory and network control, neural networks, system identification, and their applications. IEEE Membership Number : Member 07355803

TABLE I

IBM Data sets

	IBM1
Group A:	$0.33 \times (salary + commission) - 30000 > 0$
Group B:	Otherwise.
	IBM2
Group A:	$0.67 \times (salary + commission) - 5000 \times ed\_level - 20000 > 0$
Group B:	Otherwise.
	IBM3
Group A:	$0.67 \times (salary + commission) - 5000 \times ed\_level - loan/5 - 10000 > 0$
Group B:	Otherwise.

TABLE II

EXPERIMENTAL RESULTS FOR IBM DATA (PARENTHESES ARE THE SIZES OF THE DECISION TREES OF C4.5)

IBM1					
Noise power	No. of features	Classification performance (%) (C4.5/MLP)			
		MIFS-U	PCA	LDA	ICA-FX
0%	1	87.6(3)/85.8	53.0(3)/55.6	82.2(3)/84.0	96.8(3)/97.0
	2	97.8(25)/97.8	85.4(21)/85.8	–	99.6(3)/97.6
	all	97.8(27)/97.6	89.4(49)/90.2		99.6(3)/97.8
10%	1	82.0(3)/81.4	53.0(3)/56.2	81.2(3)/81.4	92.6(3)/91.8
	2	89.4(21)/90.2	81.6(37)/81.6	–	92.6(11)/92.8
	all	87.6(47)/87.8	87.4(49)/88.0	–	92.4(17)/92.2
IBM2					
Noise power	No. of features	Classification performance (%) (C4.5/MLP)			
		MIFS-U	PCA	LDA	ICA-FX
0%	1	89.4(5)/91.0	87.0(3)/87.2	96.4(3)/96.6	97.8(7)/98.0
	2	96.6(5)/97.0	89.6(13)/89.4	–	98.8(15)/98.4
	3	98.8(25)/98.8	89.6(13)/89.8	–	98.8(17)/98.8
	all	98.8(23)/98.6	93.8(33)/95.2	–	99.0(25)/98.8
10%	1	90.0(5)/90.6	87.0(3)/87.0	94.6(9)/95.2	96.2(5)/96.8
	2	94.8(13)/95.6	85.6(19)/86.0	–	94.8(13)/96.8
	3	96.0(13)/95.2	85.6(23)/85.0	–	95.2(19)/97.0
	all	95.0(21)/94.6	92.2(23)/92.4	–	95.8(29)/97.4
IBM3					
Noise power	No. of features	Classification performance (%) (C4.5/MLP)			
		MIFS-U	PCA	LDA	ICA-FX
0%	1	85.0(3)/85.0	55.4(3)/55.4	92.2(3)/92.2	93.2(3)/94.2
	2	91.2(31)/91.4	61.8(7)/63.8	–	93.6(15)/96.4
	3	90.6(29)/91.8	65.8(23)/66.0	–	97.0(3)/97.0
	4	90.2(33)/92.0	65.8(27)/66.4	–	96.8(21)/97.4
	all	92.4(65)/98.2	88.8(113)/89.6	–	97.8(39)/100.0
10%	1	84.8(3)/84.4	52.2(3)/52.2	89.0(3)/90.0	92.2(3)/93.0
	2	88.4(21)/89.6	58.8(11)/61.4	–	93.4(5)/93.2
	3	86.8(31)/88.8	63.0(11)/64.0	–	94.4(15)/94.0
	4	87.4(41)/87.0	63.0(15)/64.2	–	93.4(19)/94.2
	all	89.4(57)/92.6	79.8(103)/81.8	–	92.4(49)/93.6

TABLE III

Brief Information of the UCI Data sets Used

Name	No. of features	No. of instances	No. of classes
Sonar	60	208	2
Breast Cancer	9	699	2
Pima	8	768	2

TABLE IV

Classification performance for Sonar Target data (Parentheses are the standard deviations of 10 experiments)

No. of features	Classification performance (%) (C4.5/MLP/SVM)				
	MIFS-U	PCA	ICA	LDA	ICA-FX
1	73.1/74.8(0.32)/74.8	52.4/59.3(0.41)/58.6	65.9/67.9(0.25)/67.2	71.2/75.2(0.37)/74.1	87.5/87.3(0.17)/87.1
3	70.2/72.9(0.58)/75.5	51.0/57.9(0.42)/54.7	63.0/71.1(0.45)/69.7	–	86.1/88.1(0.37)/89.0
6	69.7/77.5(0.24)/80.8	64.9/63.8(0.72)/63.0	61.2/69.9(0.63)/70.2	–	85.6/86.4(0.42)/87.1
9	81.7/80.1(0.61)/79.9	69.7/71.2(0.67)/70.2	61.5/68.7(0.62)/68.7	–	83.2/85.0(0.83)/88.8
12	79.3/79.5(0.53)/81.3	73.1/74.0(0.64)/75.1	60.1/71.4(0.71)/71.7	–	78.2/83.4(0.49)/86.6
60	73.1/76.4(0.89)/82.7	73.1/75.5(0.96)/82.7	63.9/74.1(1.43)/77.0	–	73.1/80.0(0.78)/84.2

TABLE V

Classification performance for Breast Cancer data (Parentheses are the standard deviations of 10 experiments)

No. of features	Classification performance (%) (C4.5/MLP/SVM)				
	MIFS-U	PCA	ICA	LDA	ICA-FX
1	91.1/92.4(0.03)/92.7	85.8/86.1(0.05)/85.8	84.7/81.5(0.29)/85.1	96.8/96.6(0.07)/96.9	97.0/97.1(0.11)/97.0
2	94.7/95.8(0.17)/95.7	93.3/93.8(0.07)/94.7	87.3/85.4(0.31)/90.3	–	96.5/97.1(0.09)/97.1
3	95.8/96.2(0.15)/96.1	93.8/94.7(0.11)/95.9	89.1/85.6(0.33)/91.3	–	96.7/96.9(0.12)/96.9
6	95.0/96.1(0.08)/96.7	94.8/96.6(0.15)/96.6	90.4/90.0(0.59)/94.3	–	95.9/96.7(0.27)/96.7
9	94.5/96.4(0.13)/96.7	94.4/96.8(0.16)/96.7	91.1/93.0(0.84)/95.9	–	95.5/96.9(0.13)/96.6

TABLE VI

CLASSIFICATION PERFORMANCE FOR PIMA DATA (PARENTHESES ARE THE STANDARD DEVIATIONS OF 10 EXPERIMENTS)

No. of features	Classification performance (%) (C4.5/MLP/SVM)				
	MIFS-U	PCA	ICA	LDA	ICA-FX
1	72.8/74.1(0.19)/74.5	67.8/66.2(0.17)/66.3	69.7/71.6(0.17)/73.2	74.5/75.2(0.23)/75.6	76.0/78.6(0.11)/78.7
2	74.2/76.7(0.13)/75.8	75.0/74.4(0.23)/75.1	72.7/76.8(0.24)/76.7	–	75.2/78.2(0.25)/78.1
3	74.1/76.3(0.27)/76.8	74.2/75.1(0.23)/75.5	72.7/76.7(0.54)/76.8	–	75.7/76.7(0.18)/77.8
5	73.3/75.3(0.64)/76.6	73.7/75.2(0.39)/75.5	72.9/76.4(0.55)/77.2	–	77.2/77.8(0.38)/78.3
8	74.5/76.5(0.45)/78.1	74.5/76.6(0.31)/78.1	72.3/77.0(0.62)/77.9	–	72.9/76.7(0.48)/78.0

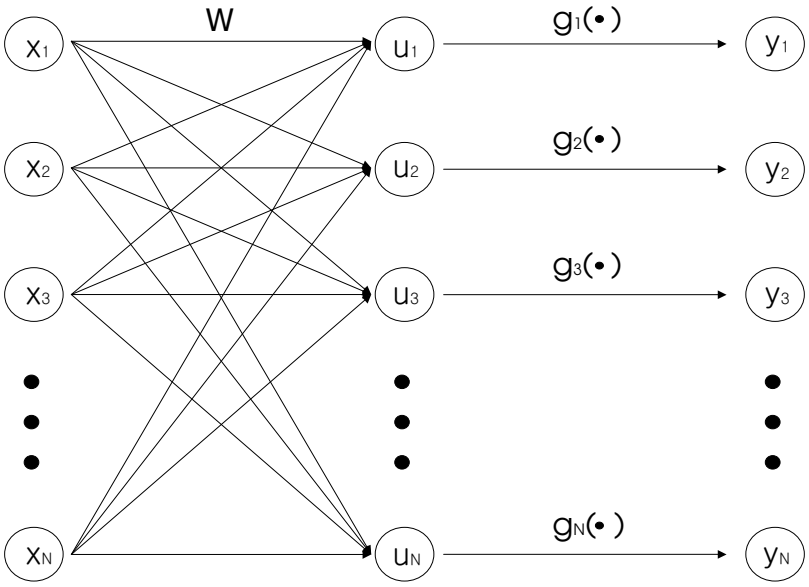


Fig. 1. Feedforward structure for ICA

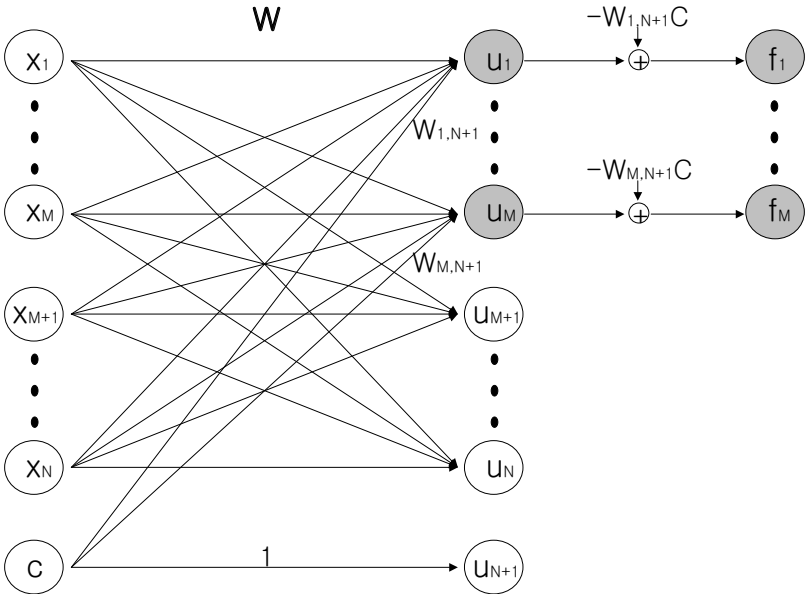
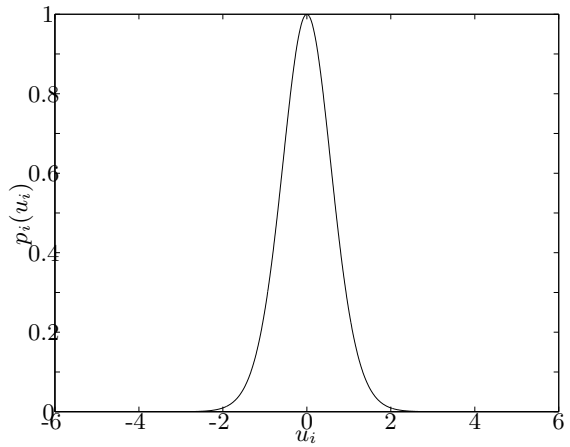
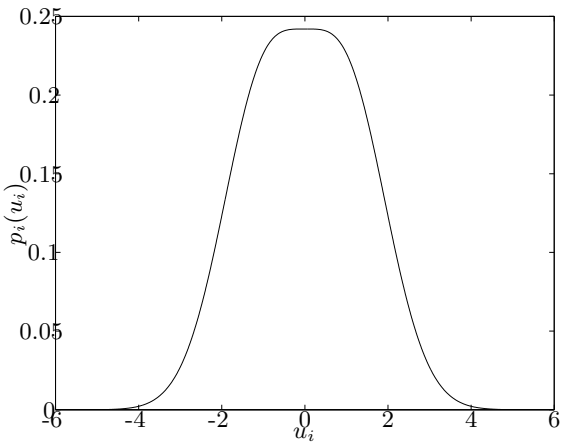


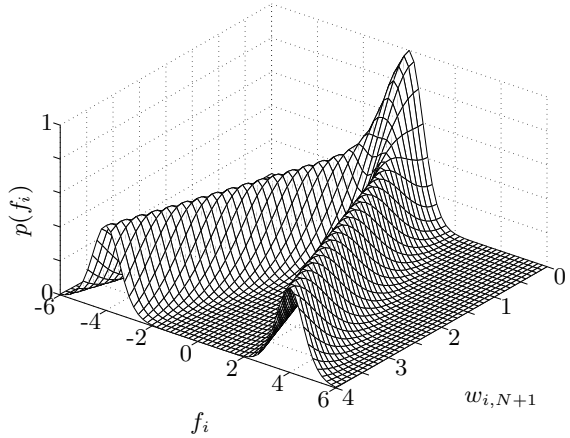
Fig. 2. Feature extraction algorithm based on ICA (ICA-FX)



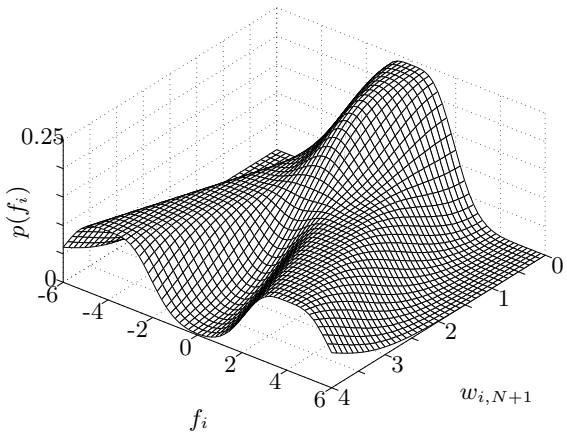
(a) Super-Gaussian density of u_i



(b) Sub-Gaussian density of u_i



(c) Density of f_i when u_i is super-Gaussian



(d) Density of f_i when u_i is sub-Gaussian

Fig. 3. Super- and sub-Gaussian densities of u_i and corresponding densities of f_i ($p_1 = p_2 = 0.5$, $c_1 = -c_2 = 1$, $\mu = 1$, and $\sigma = 1$).

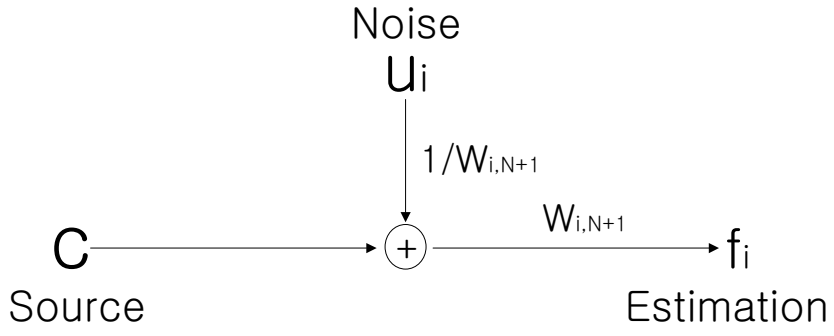


Fig. 4. Channel representation of feature extraction

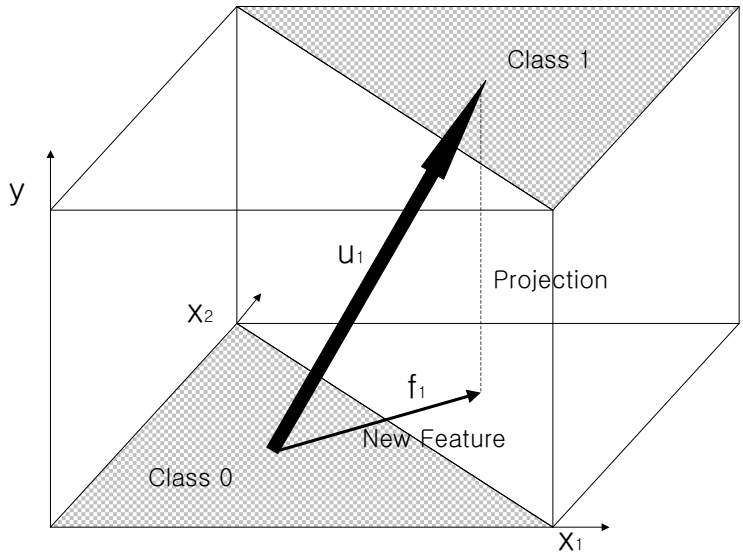
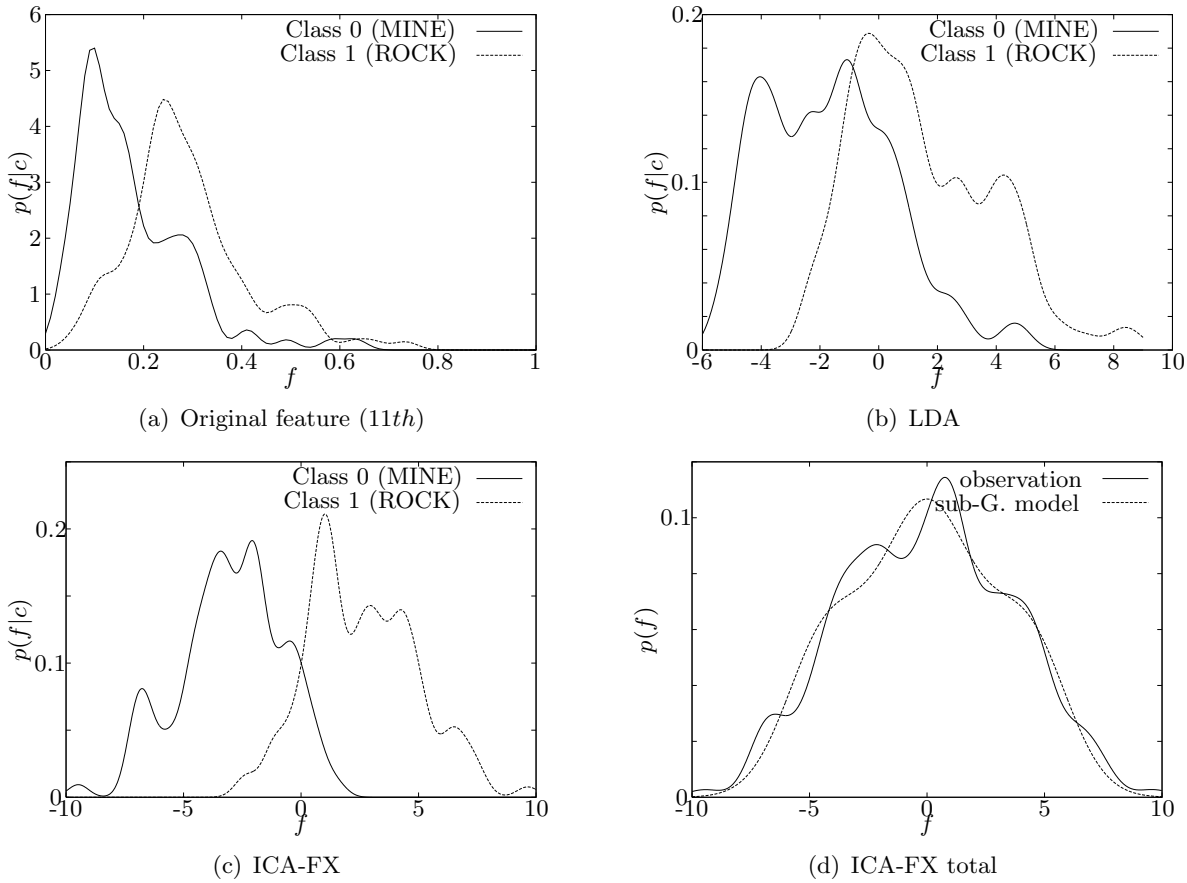


Fig. 5. Concept of ICA-FX



(a) Original feature ($11th$)

(b) LDA

(c) ICA-FX

(d) ICA-FX total

Fig. 6. Probability density estimates for a given feature (Parzen window method with window width 0.2 was used)