Feature Extraction with Weighted Samples Based on Independent Component Analysis

Nojun Kwak

Samsung Electronics, Suwon P.O. Box 105, Suwon-Si, Gyeonggi-Do, KOREA 442-742,
nojunk@ieee.org,
WWW home page: http://csl.snu.ac.kr/~nojunk

Abstract. This study investigates a new method of feature extraction for classification problems with a considerable amount of outliers. The method is a weighted version of our previous work based on the independent component analysis (ICA). In our previous work, ICA was applied to feature extraction for classification problems by including class information in the training. The resulting features contain much information on the class labels producing good classification performances. However, in many real world classification problems, it is hard to get a clean dataset and inherently, there may exist outliers or dubious data to complicate the learning process resulting in higher rates of misclassification. In addition, it is not unusual to find the samples with the same inputs to have different class labels. In this paper, Parzen window is used to estimate the correctness of the class information of a sample and the resulting class information is used for feature extraction.

1 Introduction

In this paper, the feature extraction for classification problems are dealt with and the focus is on the feature extraction by a linear transform of the original features. These methods are generally referred to as the *subspace methods* which includes principal component analysis (PCA) [1], independent component analysis (ICA) [2], Fisher's linear discriminant analysis (LDA) [3] and so on.

In our previous work, we developed ICA-FX (feature extraction based on independent component analysis) [4], a supervised feature extraction method for classification problems. Like ICA, it utilizes higher order statistics, while unlike ICA, it was developed as a supervised method in that it includes the output class information to find an appropriate feature subspace. This method is well-suited for classification problems in the aspect of constructing new features that are strongly related to output class.

In this paper, the ICA-FX is extended to incorporate the outliers and dubious data in the learning process. For a given training sample, the probability of the sample belonging to a certain class is calculated by Parzen window method [5] and this information is directly used as an input to the ICA-FX. By this preprocessing, the samples with higher class-certainty are enforced and those

with lower class-certainty are suppressed in the learning process. The proposed method is applied to an artificial dataset to show effectiveness of the method.

This paper is organized as follows. In Section 2, Parzen window method is briefly reviewed. ICA-FX, our previous feature extraction algorithm, is reviewed in Section 3 and a new method, weighted ICA-FX, is presented in Section 4. Simulation results are presented in Section 5 and conclusions follow in Section 6.

2 A Review of Parzen Window

For a given sample in a dataset, to correctly estimate in what extent the sample belongs to a class, one need to know the *pdf*s of the data. The Parzen window density estimate can be used to approximate the probability density $p(\boldsymbol{x})$ of a vector of continuous random variables $\boldsymbol{X}$ [5]. It involves the superposition of a normalized window function centered on a set of random samples. Given a set of n d-dimensional training vectors $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$, the *pdf* estimate of the Parzen window is given by

$$\hat{p}(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} \phi(\boldsymbol{x} - \boldsymbol{x}_i, h), \tag{1}$$

where $\phi(\cdot)$ is the window function and h is the window width parameter. Parzen showed that $\hat{p}(\boldsymbol{x})$ converges to the true density if $\phi(\cdot)$ and h are selected properly [5]. The window function is required to be a finite-valued non-negative density function such that

$$\int \phi(\boldsymbol{y}, h) d\boldsymbol{y} = 1, \tag{2}$$

and the width parameter is required to be a function of n such that

$$\lim_{n \to \infty} h(n) = 0, \tag{3}$$

and

$$\lim_{n \to \infty} n h^d(n) = \infty. \tag{4}$$

For window functions, the rectangular and the Gaussian window functions are commonly used. In this paper, the Gaussian window function of the following is used:

$$\phi(\boldsymbol{z}, h) = \frac{1}{(2\pi)^{d/2} h^d |\Sigma|^{1/2}} \exp(-\frac{\boldsymbol{z}^T \Sigma^{-1} \boldsymbol{z}}{2h^2}), \tag{5}$$

where Σ is a covariance matrix of a d-dimensional random vector $\boldsymbol{Z}$ whose instance is $\boldsymbol{z}$.

Figure 1 is a typical example of the Parzen window density estimate. In the figure, a Gaussian kernel is placed on top of each data point to produce the density estimate $\hat{p}(x)$.

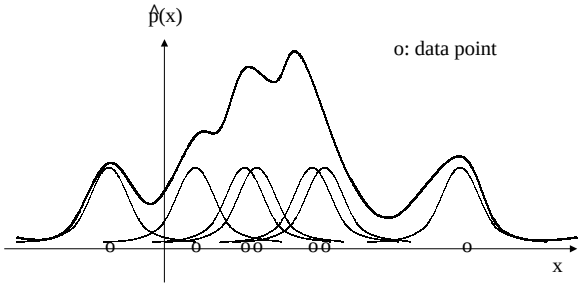


Fig. 1. An example of Parzen window density estimate

3 A Review of ICA-FX

ICA outputs a set of maximally independent vectors that are linear combinations of the observed data. Although these vectors might have some applications in such areas as blind source separation and data visualization, it is not suitable for feature extraction of classification problems, because it is the unsupervised learning that does not use class information. The effort to incorporate the standard ICA with supervised learning has been made in our previous work [4], where a new feature extraction algorithm, ICA-FX for classification problems was proposed. ICA-FX tries to solve the following problem:

(Problem statement) *Assume that there are a normalized input feature vector,* $\boldsymbol{x} = [x_1, \cdots, x_N]^T$*, and an output class,* $c \in \{c_1, \cdots, c_{N_c}\}$*. The purpose of feature extraction is to extract* $M(\leq N)$ *new features* $\boldsymbol{f_a} = [f_1, \cdots, f_M]^T$ *from* $\boldsymbol{x}$*, by a linear combination of the* x_i*'s, containing the maximum information on class* c*. Here* N_c *is the number of classes.*

The main idea of the ICA-FX is simple. It tries to apply the standard ICA algorithms to feature extraction for classification problems by making use of the class labels to produce two sets of new features; features that carry as much information on the class labels (these features will be useful for classification) as possible and the others that do not (these will be discarded). The advantage is that the general ICA algorithms can be used for feature extraction by maximizing the joint mutual information between the class labels and new features.

First, suppose $N_c(\geq 2)$ denotes the number of classes. To incorporate the class labels in the ICA structure, the discrete class labels need to be encoded into numerical variables. The 1-of-N_c scheme is used in coding classes, i.e., a class vector, $\boldsymbol{c} = [c_1, \cdots, c_{N_c}]^T$, is introduced and if a class label, c, belongs to the lth value, then c_l is activated as 1 and all the other c_i's, $i \neq l$, are set to -1. After all the training examples are presented, each $c_i, i = 1, \cdots, N_c$, is shifted in order to have zero mean and are scaled to have a unit variance.

Now consider the structure shown in Fig. 2. Here, the original feature vector $\boldsymbol{x}$ is fully connected to $\boldsymbol{u} = [u_1, \cdots, u_N]$, the class vector $\boldsymbol{c}$ is connected only

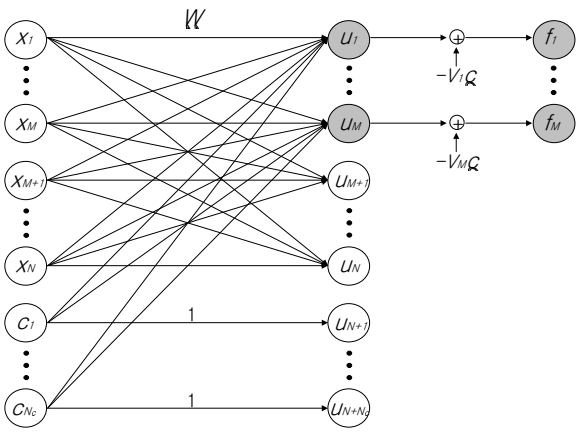


Fig. 2. Feature extraction algorithm based on ICA (ICA-FX)

to $\boldsymbol{u}_a = [u_1, \cdots, u_M]$, and $u_{N+l} = c_l$, $l = 1, \cdots, N_c$. In the figure, the weight matrix $\boldsymbol{W} \in \Re^{(N+N_c)\times(N+N_c)}$ becomes

$$\boldsymbol{W} = \left(\begin{array}{c|c} W & V \\ \hline \mathbf{0}_{N_c,N} & I_{N_c} \end{array}\right) = \left(\begin{array}{c|cc} & w_{1,N+1} & \cdots \quad w_{1,N+N_c} \\ & \vdots & \vdots \\ W & w_{M,N+1} & \cdots \quad w_{M,N+N_c} \\ & \multicolumn{2}{c}{\mathbf{0}_{N-M,N_c}} \\ \hline \mathbf{0}_{N_c,N} & \multicolumn{2}{c}{I_{N_c}} \end{array}\right). \tag{6}$$

where $W \in \Re^{N\times N}$ and $V = [V_a^T, \mathbf{0}_{N-M,N_c}^T]^T \in \Re^{N\times N_c}$. Here the first nonzero M rows of V is denoted as $V_a \in \Re^{M\times N_c}$.

In information theoretic view, the aim of feature extraction is to extract M new features $\boldsymbol{f}_a$ from the original N features, $\boldsymbol{x}$, such that $I(\boldsymbol{f}_a; c)$, the mutual information between newly extracted features $\boldsymbol{f}_a$ and the output class c, approaches $I(\boldsymbol{x}; c)$, the mutual information between the original features $\boldsymbol{x}$ and the output class c [4].

This can be satisfied if we can separate the input feature space $\boldsymbol{x}$ into two linear subspaces: one that is spanned by $\boldsymbol{f}_a = [f_1, \cdots, f_M]^T$, which contains the maximum information on the class label c, and the other spanned by $\boldsymbol{f}_b = [f_{M+1}, \cdots, f_N]^T$, which is independent of c as much as possible.

The condition for this separation can be derived as follows. If it is assumed that $\boldsymbol{W}$ is nonsingular, then $\boldsymbol{x}$ and $\boldsymbol{f} = [f_1, \cdots, f_N]^T$ span the same linear space, which can be represented with the direct sum of $\boldsymbol{f}_a$ and $\boldsymbol{f}_b$, and then by the data processing inequality [6],

$$I(\boldsymbol{x}; c) = I(W\boldsymbol{x}; c) = I(\boldsymbol{f}; c) = I(\boldsymbol{f}_a, \boldsymbol{f}_b; c) \geq I(\boldsymbol{f}_a; c). \tag{7}$$

The first equality holds because W is nonsingular. The second and the third equalities are from the definitions of $\boldsymbol{f}$, $\boldsymbol{f}_a$ and $\boldsymbol{f}_b$. In the inequality on the last line, the equality holds if $I(\boldsymbol{f}_b; c) = I(u_{M+1}, \cdots, u_N; c) = 0$.

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

To solve this problem, the feature extraction problem is interpreted in the structure of the blind source separation (BSS) problem as shown in Fig. 3. The detailed description of each step is as follows:

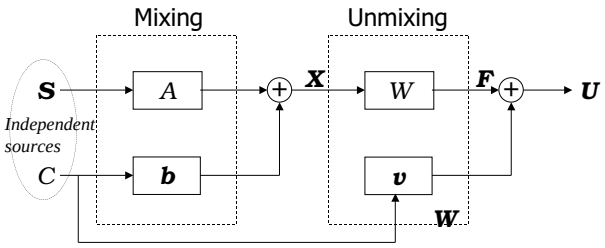


Fig. 3. Interpretation of Feature Extraction in the BSS structure

(Mixing) *Assume that there are N independent sources $\boldsymbol{s} = [s_1, \cdots, s_N]^T$ which are also independent of the class label c. Assume also that the observed feature vector $\boldsymbol{x}$ is a linear combination of the sources $\boldsymbol{s}$ and $\boldsymbol{c}$ with the mixing matrix $A \in \Re^{N \times N}$ and $B \in \Re^{N \times N_c}$; i.e.,*

$$\boldsymbol{x} = A\boldsymbol{s} + B\boldsymbol{c}. \tag{8}$$

(Unmixing) *The unmixing stage is slightly different from the BSS problem as shown in Fig. 2. In the figure, the unmixing equation becomes*

$$\boldsymbol{u} = W\boldsymbol{x} + V\boldsymbol{c}. \tag{9}$$

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

$$\boldsymbol{e} \triangleq \Lambda\Pi\boldsymbol{s} \tag{10}$$

where Λ is a diagonal matrix corresponding to an appropriate scale and Π is a permutation matrix. The u_i's ($i = 1, \cdots, N$) are then independent of the class label c by the assumption. Among the elements of $\boldsymbol{f} = W\boldsymbol{x} (= \boldsymbol{u} - V\boldsymbol{c})$, $\boldsymbol{f}_b = [f_{M+1}, \cdots, f_N]^T$ will be independent of c because the ith row of V, $V_i = [w_{i,N+1}, \cdots, w_{i,N+N_c}] = \boldsymbol{0}$ and $f_i = u_i$ for $i = M+1, \cdots, N$. Therefore, the $M(< N)$ dimensional new feature vector $\boldsymbol{f}_a$ can be extracted by a linear transformation of $\boldsymbol{x}$ containing the most information on the class if the relation $\boldsymbol{u} = \boldsymbol{e}$ holds.

The learning rule for the ICA-FX is obtained in a similar way as that of ICA using the MLE approach as follows.

If it is assumed that $\boldsymbol{u} = [u_1, \cdots, u_N]^T$ is a linear combination of the source $\boldsymbol{s}$; i.e., it is made equal to $\boldsymbol{e}$, a scaled and permutated version of the source, $\boldsymbol{s}$, as in (10), and that each element of $\boldsymbol{u}$ is independent of the other elements of $\boldsymbol{u}$, which is also independent of the class vector $\boldsymbol{c}$, the log likelihood of the data for a given $\boldsymbol{W}$ becomes the following:

$$L(\boldsymbol{u}, \boldsymbol{c}|\boldsymbol{W}) = \log|\det \boldsymbol{W}| + \sum_{i=1}^{N} \log p_i(u_i) + \log p(\boldsymbol{c}) \tag{11}$$

because

$$p(\boldsymbol{x}, \boldsymbol{c}|\boldsymbol{W}) = |\det \boldsymbol{W}|\ p(\boldsymbol{u}, \boldsymbol{c}) = |\det \boldsymbol{W}|\ \prod_{i=1}^{N} p_i(u_i)\ p(\boldsymbol{c}). \tag{12}$$

Now, L can be maximized, and this can be achieved by the steepest ascent method. Because the last term in (11) is a constant, differentiating (11) with respect to $\boldsymbol{W}$ leads to

$$\begin{aligned} \frac{\partial L}{\partial w_{i,j}} &= \frac{adj(w_{j,i})}{|\det \boldsymbol{W}|} - \varphi_i(u_i) x_j \qquad 1 \le i, j \le N \\ \frac{\partial L}{\partial w_{i,N+j}} &= -\varphi_i(u_i) c_j \qquad 1 \le i \le M, 1 \le j \le N_c \end{aligned} \tag{13}$$

where $adj(\cdot)$ is adjoint and $\varphi_i(u_i) = -\frac{dp_i(u_i)}{du_i}/p_i(u_i)$. Note that each c_i has binary numerical values depending on the class label c.

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

$$\begin{aligned} \Delta W &\propto\ W^{-T} - \boldsymbol{\varphi}(\boldsymbol{u})\boldsymbol{x}^T \\ \Delta V_a &\propto\ -\boldsymbol{\varphi}(\boldsymbol{u}_a)\boldsymbol{c}^T. \end{aligned} \tag{14}$$

Here $\boldsymbol{\varphi}(\boldsymbol{u}) \triangleq [\varphi_1(u_1), \cdots, \varphi_N(u_N)]^T$ and $\boldsymbol{\varphi}(\boldsymbol{u}_a) \triangleq [\varphi_1(u_1), \cdots, \varphi_M(u_M)]^T$.

Applying a natural gradient on updating W, by multiplying $W^T W$ on the right side of the first equation of (**??**), the following is obtained.

$$\begin{aligned} W^{(t+1)} &= W^{(t)} + \mu_1 [I_N - \boldsymbol{\varphi}(\boldsymbol{u})\boldsymbol{f}^T] W^{(t)} \\ V_a^{(t+1)} &= V_a^{(t)} - \mu_2 \boldsymbol{\varphi}(\boldsymbol{u}_a)\boldsymbol{c}^T. \end{aligned} \tag{15}$$

Here μ_1 and μ_2 are the learning rates that can be set differently. By this weight update rule, the resulting u_i's will have a good chance of fulfilling the assumption that u_i's are not only independent of one another but also independent of the class label c.

Note that the learning rule for W is the same as the original ICA learning rule [2], and also note that $\boldsymbol{f}_a$ corresponds to the first M elements of $W\boldsymbol{x}$. Therefore, the optimal features $\boldsymbol{f}_a$ can be extracted by the proposed algorithm when it finds the optimal solution for W by (15).

4 Weighted ICA-FX

In ICA-FX presented in the above section, the 1-of-N_c scheme was used to code the discrete class labels into numerical ones, but in many real world problems the same sample may be classified as either one or another class with probability. In addition, the training data may contain incorrect class information resulting errors in classification. This problem may be solved if the probabilistic coding scheme is used for coding the discrete class information into numerical values. That is, suppose there are 3 classes and a training sample says that it belongs to *class 1*. Because the class information of this sample may or may not be correct, instead of using $(1, 0, 0)$ for coding the class of this sample, probabilistic coding such as $(0.7, 0.1, 0.2)$ using the other training data can be used to train ICA-FX. This is done if we know the conditional distribution of classes for a given dataset $p(c|\boldsymbol{x})$.

For this purpose, Parzen window presented in Section 2, is used to estimate the probability that the sample belongs to either *class 1*, *class 2* or *class 3* as follows.

By the Bayesian rule, the conditional probability $p(c|\boldsymbol{x})$ can be written as

$$p(c|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|c)p(c)}{p(\boldsymbol{x})}. \tag{16}$$

If the class has N_c values, say $1, 2, \cdots, N_c$, the estimate of the conditional *pdf* $\hat{p}(\boldsymbol{x}|c)$ of each class is obtained using the Parzen window method as

$$\hat{p}(\boldsymbol{x}|c) = \frac{1}{n_c} \sum_{i \in I_c} \phi(\boldsymbol{x} - \boldsymbol{x}_i, h), \tag{17}$$

where $c = 1, \cdots, N_c$; n_c is the number of the training examples belonging to class c; and I_c is the set of indices of the training examples belonging to class c. Because the summation of the conditional probability equals one, i.e.,

$$\sum_{k=1}^{N_c} p(k|\boldsymbol{x}) = 1,$$

the conditional probability $p(c|\boldsymbol{x})$ is

$$p(c|\boldsymbol{x}) = \frac{p(c|\boldsymbol{x})}{\sum_{k=1}^{N_c} p(k|\boldsymbol{x})} = \frac{p(c)p(\boldsymbol{x}|c)}{\sum_{k=1}^{N_c} p(k)p(\boldsymbol{x}|k)}.$$

The second equality is by the Bayesian rule (16). Using (17), the estimate of the conditional probability becomes

$$\hat{p}(c|\boldsymbol{x}) = \frac{\sum_{i \in I_c} \phi(\boldsymbol{x} - \boldsymbol{x}_i, h_c)}{\sum_{k=1}^{N_c} \sum_{i \in I_k} \phi(\boldsymbol{x} - \boldsymbol{x}_i, h_k)}, \tag{18}$$

where h_c and h_k are the class specific window width parameters. Here $\hat{p}(k) = n_k/n$ is used instead of the true density $p(k)$.

If the Gaussian window function (5) is used with the same window width parameter and the same covariance matrix for each class, (18) becomes

$$\hat{p}(c|\boldsymbol{x}) = \frac{\sum_{i \in I_c} \exp(-\frac{(\boldsymbol{x}-\boldsymbol{x}_i)^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{x}_i)}{2h^2})}{\sum_{k=1}^{N_c} \sum_{i \in I_k} \exp(-\frac{(\boldsymbol{x}-\boldsymbol{x}_i)^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{x}_i)}{2h^2})}. \tag{19}$$

Note that for multi-class classification problems, there may not be enough samples such that the error for the estimate of class specific covariance matrix can be large. Thus, the same covariance matrix is used for each class throughout this paper.

Using $\hat{p}(c|\boldsymbol{x})$ obtained above, the class vector $\boldsymbol{c}$ in Section 3 becomes probabilistic depending on the whole dataset. And this can be used in training ICA-FX directly. The advantage of this coding scheme over 1-of-N_c scheme is that the class information of a sample is affected by its neighboring samples and it becomes more tolerant to outliers. This smoothing process acts as giving more (less) weights on samples whose class information is trustworthy (uncertain). From now on, the proposed algorithm will be referred to as the weighted ICA-FX (wICA-FX).

5 Simulation Results

In this section, the performance of wICA-FX is compared with those of other methods. Consider the simple problem of the following:

Suppose we have two independent input features x_1 and x_2 uniformly distributed on [-0.5,0.5] for a binary classification, and the output class c is determined as follows:

$$c = \begin{cases} 0 & \text{if } x_1 + 3x_2 < 0 \\ 1 & \text{if } x_1 + 3x_2 \geq 0. \end{cases}$$

For this problem, 5 datasets were generated where the class c was randomly flipped with probability of 0 to 0.4. Each dataset contains 500 samples on which PCA, LDA, ICA, ICA-FX and wICA-FX were performed. These feature extraction methods were tested on a separate test dataset with no flip of class information.

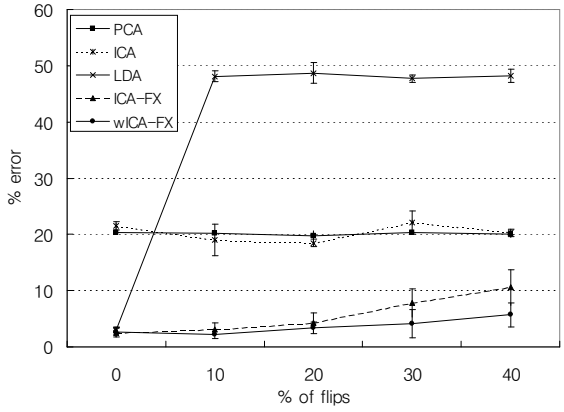
Table 1 is the classification performances of various feature extraction methods on these datasets. One feature is extracted with each method. Averages of 10 experiments with standard deviations are reported here. Standard multi-layer perceptron (MLP) with one hidden layer was used for the classification. Three hidden nodes were used with learning rate of 0.02 and momentum of 0.9. The number of iterations was set to 100. In wICA-FX, h was set to $\frac{1}{\log_{10} n}$ as in [7], where n is the number of training samples.

In the table, the performances of LDA, ICA-FX, and wICA-FX are almost the same when there are no flipped classes. As the number of flipped samples

increases, the error rates of wICA-FX increase more slowly than those of ICA-FX. Comparing to ICA-FX and wICA-FX, the error rates of LDA suddenly jump to 48% when only 10% of the samples are flipped. Note that the error rates of PCA and ICA stays the same around 20 % because these are unsupervised learning methods.

Table 1. Classification performance for the simple dataset (Averages of 10 experiments. Numbers in the parentheses are the standard deviations.)

% of flips	Classification error (%) (MLP)				
	PCA	ICA	LDA	ICA-FX	**wICA-FX**
0	20.41	21.53	2.90	2.54	**2.64**
	(0.32)	(0.70)	(0.42)	(0.84)	(0.86)
10	20.22	19.06	48.10	3.16	**2.28**
	(0.28)	(2.82)	(0.98)	(1.15)	(0.74)
20	19.74	18.67	48.71	4.24	**3.42**
	(0.98)	(0.71)	(1.83)	(1.74)	(1.05)
30	20.30	22.18	47.72	7.82	**4.16**
	(0.14)	(2.12)	(0.71)	(2.51)	(2.47)
40	20.02	20.37	48.21	10.56	**5.68**
	(0.56)	(0.70)	(1.13)	(3.21)	(2.09)



6 Conclusions

This study investigates a new method of feature extraction for classification problems with a considerable amount of outliers. In our previous work ICA-FX, class information was added 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. However in many real world classification problems, it is hard to get a clean dataset and inherently, there may exist outliers or dubious data to complicate the learning process resulting errors in classification. In addition, a sample may be classified as either one or another class with probability. The proposed method focuses on this problem and it is a weighted version of ICA-FX. Parzen window is used to estimate the correctness of the class information of a sample and the resulting class information is used to code the class in ICA-FX. The advantage of this coding scheme over 1-of-N_c scheme is that the class information of a sample is affected by its neighboring samples, thus becomes more tolerant to outliers. This smoothing process acts as giving more (less) weights on samples whose class information is trustworthy (uncertain). Experimental result on the simple artificial dataset shows that the wICA-FX is very effective in dealing with the incorrect class information.

References

1. I.T. Joliffe, *Principal Component Analysis*, Springer-Verlag, 1986.
2. 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.
3. K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, second edition, 1990.
4. N. Kwak and C.-H. Choi, "Feature extraction based on ica for binary classification problems," *IEEE Trans. on Knowledge and Data Engineering*, vol. 15, no. 6, pp. 1374–1388, Nov. 2003.
5. E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statistics*, vol. 33, pp. 1065–1076, Sept. 1962.
6. T.M. Cover and J.A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991.
7. C. Meilhac and C. Nastar, "Relevance feedback and catagory search in image databases," in *Proc. IEEE Int'l Conf. on Content-based Access of Video and Image databases*, Florence, Italy, June 1999.