Fast Independent Component Analysis in Kernel Feature Spaces

András Kocsor and János Csirik

Research Group on Artificial Intelligence
of the Hungarian Academy of Sciences and the University of Szeged
H-6720 Szeged, Aradi vértanúk tere 1., Hungary
{kocsor, csirik}@inf.u-szeged.hu
http://www.inf.u-szeged.hu/speech

Abstract. It is common practice to apply linear or nonlinear feature extraction methods before classification. Usually linear methods are faster and simpler than nonlinear ones but an idea successfully employed in the nonlinearization of Support Vector Machines permits a simple and effective extension of several statistical methods to their nonlinear counterparts. In this paper we follow this general nonlinearization approach in the context of Independent Component Analysis, which is a general purpose statistical method for blind source separation and feature extraction. In addition, nonlinearized formulae are furnished along with an illustration of the usefulness of the proposed method as an unsupervised feature extractor for the classification of Hungarian phonemes.

KeyWords. feature extraction, kernel methods, Independent Component Analysis, FastICA, phoneme classification

1 Introduction

Feature extraction methods, whether in linear or a nonlinear form, produce preprocessing transformations of high dimensional input data, which may increase the overall performance of classifiers in many real world applications. These algorithms also permit the restriction of the entire input space to a subspace of lower dimensionality. In general, experience has shown that dimensionality reduction has a favorable effect on the classification performance, i.e. reducing superfluous features which can disturb the goal of separation.

In this study Independent Component Analysis will be derived in a nonlinearized form, where the method of nonlinearization was performed by employing the so-called "kernel-idea" [11]. This notion can be traced back to the potential function method [1], and its renewed use in the ubiquitous Support Vector Machine [4], [21].

Without loss of generality we shall assume that as a realization of multivariate random variables, there are m-dimensional real attribute vectors in a compact set $\mathcal{X}$ over $\mathbb{R}^m$ describing objects in a certain domain, and that we have a

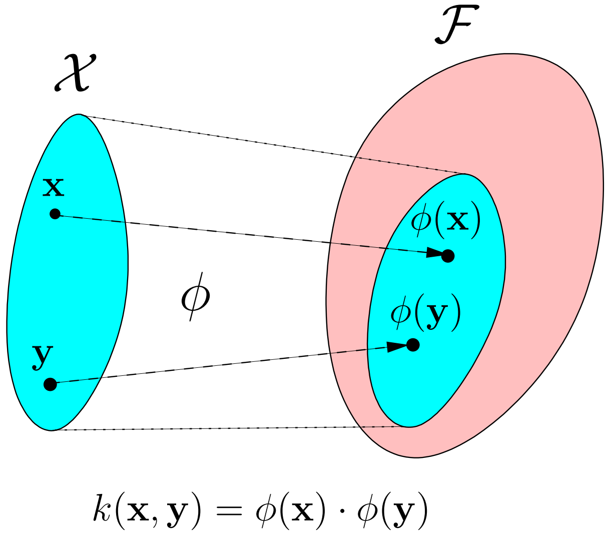

Fig. 1. The "kernel-idea". $\mathcal{F}$ is the closure of the linear span of the mapped data. The dot product in the kernel feature space $\mathcal{F}$ is defined implicitly. The dot product of $\sum_{i=1}^{n} \alpha_i \phi(\mathbf{x_i})$ and $\sum_{i=1}^{n} \beta_i \phi(\mathbf{x_i})$ is $\sum_{i,j} \alpha_i \beta_j k(\mathbf{x_i}, \mathbf{x_j})$.

finite $m \times n$ sample matrix $X = [\mathbf{x_1}, \ldots, \mathbf{x_n}]$ containing n random observations. The aim of Independent Component Analysis (ICA) is to linearly transform the sample matrix X into components that are as independent as possible. The definition of independence of the components can be viewed in different ways. In [8] and [9] Hyvärinen proposed a new concept and a new method (i.e. FastICA) that extends Comon's information theoretic ICA approach [6] with a new family of contrast functions. FastICA is a fast approximate Newton iteration procedure (the convergence is at least quadratic) for the optimization of the negentropy approximant (see definition later), which serves as a measure for selecting new independent components. Fortunately this method can be reexpressed as its input is $K = X^\top X$ instead of X, where the $n \times n$ symmetric matrix K is the pairwise combination of dot products of the sample ($K = [\mathbf{x_i} \cdot \mathbf{x_j}]_{ij}$). Now let the dot product be implicitly defined (Fig. 1) by the kernel function k in some finite or infinite dimensional feature space $\mathcal{F}$ with associated transformation ϕ:

$$k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y}). \tag{1}$$

Going the other way, constructing an appropriate kernel function (i.e. where such a function ϕ exists) is a non-trivial problem, but there are many good suggestions about the sorts of kernel functions which might be adopted along with some background theory [21], [5]. However, the two most popular kernels are the following:

$$\text{Polynomial kernel:} \quad k_1(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y})^d, \qquad d \in \mathbb{N}, \tag{2}$$

$$\text{Gaussian RBF kernel:} \quad k_2(\mathbf{x},\mathbf{y}) = \exp\left(-||\mathbf{x}-\mathbf{y}||^2/r\right), \qquad r \in \mathbb{R}_+. \tag{3}$$

For a given kernel function the pair $(\phi, \dim \mathcal{F})$ is not always unique and for the kernels k_1 and k_2 the following statements hold:

i) If the dot product is computed as a polynomial kernel, then the dimension of the feature space is at least $\binom{m+d-1}{d}$.

ii) The dot product using the Gaussian RBF kernel induces infinite dimension feature spaces.

As the input of FastICA is represented only by dot products, matrix K is easily redefinable by

$$K = [k(\mathbf{x_i}, \mathbf{x_j})]_{ij}. \tag{4}$$

With this substitution FastICA produces a linear transformation matrix in the kernel feature space $\mathcal{F}$, but now this is no longer a linear transformation of the input data owing to the nonlinearity of ϕ. Still, dot products in $\mathcal{F}$ computed with kernels offer a fast implicit access to this space that in turn leads to a low complexity nonlinear extractor. If we have a low-complexity (perhaps linear) kernel function the dot product $\phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j})$ can also be computed with fewer operations (e.g. $O(m)$) whether or not $\phi(\mathbf{x})$ is infinite in dimension.

Using this general schema various feature extraction methods such as Principal Component Analysis (the first generalization right after SVM, proposed by Schölkopf et al.) [19], [15], [13], Linear Discriminant Analysis [16], [18], [20] and Independent Component Analysis have already been nonlinearized. Hopefully other statistical methods will uncover their nonlinear counterparts in the near future.

In the subsequent section we will review the standard Independent Component Analysis and FastICA algorithms. Afterwards we will reformulate ICA in such a way that its input is the dot product of the input data, and then this basic operation will be replaced by kernel functions. The final part of the paper will discuss the results of experiments on the phoneme classification followed by some concluding remarks.

2 Independent Component Analysis

Independent Component Analysis [6], [8], [9], [7] is a general purpose statistical method that originally arose from the study of blind source separation (BSS). A typical BSS problem is the cocktail-party problem where several people are speaking simultaneously in the same room and several microphones record a mixture of speech signals. The task is to separate the voices of different speakers using the recorded samples. Another application of ICA is feature extraction, where the aim is to linearly transform the input data into uncorrelated components, along which the distribution of the sample set is the least Gaussian. The reason for this is that along these directions the data is supposedly easier to classify. For optimal selection of the independent directions several contrast function were defined using approximately equivalent approaches. Here we follow the way proposed by Hyvärinen [8], [9], [7]. Generally speaking, we expect these functions to be non-negative and have a zero value for a Gaussian distribution. Negentropy is a useful measure having just this property, used for assessing non-Gaussianity (i.e. the least Gaussianity). Since obtaining this quantity via its definition is computationally rather difficult, a simple easily-computable approximation is normally employed. The negentropy of a variable η with zero mean and unit variance is estimated by the formula

$$J_G(\eta) \approx \left(E\{G(\eta)\} - E\{G(\nu)\}\right)^2 \tag{5}$$

where $G() : \mathbb{R} \rightarrow \mathbb{R}$ is an appropriate nonquadratic function, E denotes the expected value and ν is a standardized Gaussian variable. The following three choices of $G(\eta)$ are conventionally used: η^4, $\log(\cosh(\eta))$ or $-\exp(-\eta^2/2)$. It should be noticed that in (5) the expected value of $G(\nu)$ is a constant, its value only depending on the selected contrast function (e.g. $E(G_1(\nu)) = 3$). Hyvärinen recently proposed a fast iterative algorithm called FastICA for the selection of the new base vectors of the linearly transformed space. The goodness of a new direction $\mathbf{a}$ is measured by the following function, where η is replaced with $\boldsymbol{a} \cdot \mathbf{x}$ in the negentropy approximant (5):

$$\tau_G(\boldsymbol{a}) = \left(E\{G(\boldsymbol{a} \cdot \mathbf{x})) - E\{G(\nu)\}\right)^2 . \tag{6}$$

As a matter of fact, FastICA is an approximate Newton iteration procedure for the local optimization of the function $\tau_G(\boldsymbol{a})$. Before running FastICA, however, the raw input data X must first be preprocessed – by centering and whitening it. Between centering and whitening we may, perhaps, also apply a deviance normalization because the standardized data used as an input for the whitening sometimes improves the efficiency of the FastICA algorithm. However, we should mention here there are many other iterative methods for performing Independent Component Analysis. Some of these (similar to FastICA) do require centering and whitening, while others do not. In general, experience has taught us that all these algorithms should converge faster on centered and whitened data, even those which do not really require it.

Centering. An essential step is to shift the original sample set $\mathbf{x_1}, \ldots, \mathbf{x_n}$ with its mean $\boldsymbol{\mu} = E\{\mathbf{x}\}$, to obtain data $\mathbf{x_1'} = \mathbf{x_1} - \boldsymbol{\mu}, \ldots, \mathbf{x_n'} = \mathbf{x_n} - \boldsymbol{\mu}$, with a mean of zero.

Whitening. The goal of this step is to transform the centered samples $\mathbf{x_1'}, \ldots, \mathbf{x_n'}$ via an orthogonal transformation Q into a space where the covariance matrix $\hat{C} = E\{\hat{\mathbf{x}}\hat{\mathbf{x}}^\top\}$ of the points $\hat{\mathbf{x}}_\mathbf{1} = Q^\top \mathbf{x_1'}, \ldots, \hat{\mathbf{x}}_\mathbf{n} = Q^\top \mathbf{x_n'}$ is the unit matrix. Since the standard principal component analysis [10] transforms the covariance matrix into a diagonal form, the elements in the diagonal being the eigenvalues of the original covariance matrix $C' = E\{\mathbf{x}'\mathbf{x}'^\top\}$, it only remains to transform each

diagonal element to one. It is readily seen that the sample covariance matrix C' is symmetric positive semidefinite, so the eigenvectors are orthogonal and the corresponding real eigenvalues are nonnegative. If we then further assume that the eigenpairs of C' are $(\mathbf{c_1}, \lambda_1), \dots, (\mathbf{c_m}, \lambda_m)$ and $\lambda_1 \geq \dots \geq \lambda_m$, the transformation matrix Q will take the form $[\mathbf{c_1}\lambda_1^{-1/2}, \dots, \mathbf{c_k}\lambda_k^{-1/2}]$. If k is less than m a dimensionality reduction is employed.
Properties of the preprocessing stage. Firstly, after centering and whitening for every normalized $\boldsymbol{a}$ the mean of $\boldsymbol{a} \cdot \hat{\mathbf{x}}_\mathbf{1}, \dots, \boldsymbol{a} \cdot \hat{\mathbf{x}}_\mathbf{n}$ is zero, and its variance is one. Actually we need this since (5) requires that η has a zero mean and variance of one hence, with the substitution $\eta = \boldsymbol{a} \cdot \hat{\mathbf{x}}$, the projected data $\boldsymbol{a} \cdot \hat{\mathbf{x}}$ must also have this property. Secondly, for any matrix W the covariance matrix C_W of the transformed points $W\hat{\mathbf{x}}_\mathbf{1}, \dots, W\hat{\mathbf{x}}_\mathbf{n}$ will remain a unit matrix if and only if W is orthogonal, since

$$C_W = E\{W\hat{\mathbf{x}}(W\hat{\mathbf{x}})^\top\} = WE\{\hat{\mathbf{x}}\hat{\mathbf{x}}^\top\}W^\top = WIW^\top = WW^\top \tag{7}$$

FastICA. After preprocessing, this method finds a new orthogonal base W for the preprocessed data, where the values of the non-Gaussianity measure τ_G for the base vectors are large[1]. The following pseudo-code give further details[2]:

```
% The input for this algorithm is the sample matrix $X$ and the
% nonlinear function $G$, while the output is the transformation
% matrix $W$. The first and second order derivatives of $G$ are
% denoted by $G^{'}$ and $G^{''}$. $(W_iW_i^\top)^{-1/2}W_i$ is a symmetric
% decorrelation, where $(W_iW_i^\top)^{-1/2}$ can be obtained from its
% eigenvalue decomposition. If $W_iW_i^\top = EDE^\top$, then
% $(W_iW_i^\top)^{-1/2}$ is equal to $ED^{-1/2}E^\top$.
procedure FastICA($X, G$);
      $\boldsymbol{\mu} = E\{\mathbf{x}\}$;  $\mathbf{x}' = \mathbf{x} - \boldsymbol{\mu}$;  $\hat{\mathbf{x}} = \mathbf{Q}^\top\mathbf{x}'$;  % centering & whitening
      Let $W_0$ be a random $m \times m$ orthogonal matrix;
      $W_0 = (W_0W_0^\top)^{-1/2}W_0$;
      $i = 0$;
      While $W$ has not converged;
            for $j = 1$ to $m$
                  let $\mathbf{s_j}$ be the $jth$ raw vector of $W_i$;
                  $\mathbf{w_j} = E\{\hat{\mathbf{x}}G^{'}(\mathbf{s_j} \cdot \hat{\mathbf{x}})\} - E\{G^{''}(\mathbf{s_j} \cdot \hat{\mathbf{x}})\}\mathbf{s_j}$;
            end;
            $i = i + 1$;
            $W_i = [\mathbf{w_1}, \dots, \mathbf{w_p}]^\top$;
            $W_i = (W_iW_i^\top)^{-1/2}W_i$;
      do
End procedure
```

[1] Note that since the data remains whitened after an orthogonal transformation, ICA can be considered an extension of PCA.

[2] `MatLab` code available in [7].

Transformation of test vectors. For an arbitrary test vector $\mathbf{y} \in \mathcal{X}$ the transformation can be done using $\mathbf{y}^* = WQ^\top(\mathbf{y} - \boldsymbol{\mu})$. Here W denotes the orthogonal transformation matrix we obtained as the output from FastICA, while Q is the matrix obtained from whitening.

3 Independent Component Analysis with Kernels

In this section we derive the kernel counterpart of *FastICA*. To this end, let the inner product be implicitly defined by the kernel function k in $\mathcal{F}$ with associated transformation ϕ. Now we need only extend nonlinearly the centering and whitening of the data, since after nonlinearizing $Q^\top(\mathbf{y} - \boldsymbol{\mu})$ we get data in $\mathcal{F}$ thus the nonlinearization of the iterative section becomes superfluous.
Centering in $\mathcal{F}$. We shift the data $\phi(\mathbf{x_1}), \ldots, \Phi(\mathbf{x_n})$ with its mean $\boldsymbol{\mu}^\phi$, to obtain data $\phi'(\mathbf{x_1}), \ldots, \phi'(\mathbf{x_n})$ with a mean of zero:

$$\phi'(\mathbf{x_1}) = \phi(\mathbf{x_1}) - \boldsymbol{\mu}^\Phi, \ldots, \phi'(\mathbf{x_n}) = \phi(\mathbf{x_n}) - \boldsymbol{\mu}^\phi, \qquad \boldsymbol{\mu}^\phi = \frac{1}{n}\sum_{i=1}^{n} \phi(\mathbf{x_i}) \tag{8}$$

Whitening in $\mathcal{F}$. Much like that in linear ICA, the goal of this step is to find a transformation $Q_{\hat{\phi}}$ such that the covariance matrix

$$C^{\hat{\phi}} = \frac{1}{n}\sum_{i=1}^{n} \hat{\phi}(\mathbf{x_i})\hat{\phi}(\mathbf{x_i})^\top \tag{9}$$

of the sample $\hat{\phi}(\mathbf{x_1}) = Q_{\hat{\phi}}{}^\top \phi'(\mathbf{x_1})$, ..., $\hat{\phi}(\mathbf{x_n}) = Q_{\hat{\phi}}{}^\top \phi'(\mathbf{x_n})$ is a unit matrix. As we saw earlier the column vectors of $Q_{\hat{\phi}}$ are the weighted eigenvectors of the positive semidefinite matrix $C^{\hat{\phi}}$. Because this eigen-problem is equivalent to determining the stationary points of the Rayleigh Quotient

$$\frac{\mathbf{a}^\top C^{\hat{\phi}}\mathbf{a}}{\mathbf{a}^\top\mathbf{a}}, \quad \mathbf{0} \neq \mathbf{a} \in \mathcal{F}, \tag{10}$$

this formula will be rearranged as an expression of dot products of the input data. Owing to the special form of $\mathbf{C}^{\hat{\phi}}$ we suppose that when we search for stationary points, $\mathbf{a}$ has the form

$$\mathbf{a} = \sum_{i=1}^{n} \alpha_i \hat{\phi}(\mathbf{x_i}). \tag{11}$$

We may arrive at this assumption in various ways, e.g. by decomposing an arbitrary vector $\mathbf{a}$ into $\mathbf{a_1} + \mathbf{a_2}$, where $\mathbf{a_1}$ is that component of $\mathbf{a}$ which falls in $SPAN(\hat{\phi}(\mathbf{x_1}), \ldots, \hat{\phi}(\mathbf{x_n}))$, while $\mathbf{a_2}$ is the component perpendicular to it. Then from the derivation of (10) we see that $\mathbf{a_2} \cdot \mathbf{a_2} = 0$ for the stationary points.

The following formulas give the Rayleigh Quotient as a function of $\boldsymbol{\alpha}$ and $k(\mathbf{x_i}, \mathbf{x_j})$:

$$\frac{\mathbf{a}^\top C^{\hat{\phi}} \mathbf{a}}{\mathbf{a}^\top \mathbf{a}} = \frac{\left(\sum_{t=1}^n \alpha_t \hat{\phi}(\mathbf{x_t})^\top\right) C^{\hat{\phi}} \left(\sum_{k=1}^n \alpha_k \hat{\phi}(\mathbf{x_k})\right)}{\left(\sum_{t=1}^n \alpha_t \hat{\phi}(\mathbf{x_t})^\top\right) \left(\sum_{k=1}^n \alpha_k \hat{\phi}(\mathbf{x_k})\right)} = \frac{\boldsymbol{\alpha}^\top \frac{1}{n} \hat{K} \hat{K} \boldsymbol{\alpha}}{\boldsymbol{\alpha}^\top \hat{K} \boldsymbol{\alpha}}, \tag{12}$$

where

$$\begin{aligned} \hat{K}_{tk} = {} & \left(\phi(\mathbf{x_t})^\top - \left(\tfrac{1}{n} \textstyle\sum_{i=1}^n \phi(\mathbf{x_i})^\top\right)\right) \left(\phi(\mathbf{x_k}) - \left(\tfrac{1}{n} \textstyle\sum_{i=1}^n \phi(\mathbf{x_i})\right)\right) = \\ & k(\mathbf{x_t}, \mathbf{x_k}) - \left(\tfrac{1}{n} \textstyle\sum_{i=1}^n \left(k(\mathbf{x_i}, \mathbf{x_k}) + k(\mathbf{x_t}, \mathbf{x_i})\right)\right) + \tfrac{1}{n^2} \textstyle\sum_{i=1}^n \sum_{j=1}^n k(\mathbf{x_i}, \mathbf{x_j}) \end{aligned} \tag{13}$$

From differentiating (12) with respect to $\boldsymbol{\alpha}$ we see that the stationary points are the solution vectors of the general eigenvalue problem $\frac{1}{n} \hat{K} \hat{K} \boldsymbol{\alpha} = \lambda \hat{K} \boldsymbol{\alpha}$, which in this case is obviously equivalent to the problem $\frac{1}{n} \hat{K} \boldsymbol{\alpha} = \lambda \boldsymbol{\alpha}$. Moreover, since $k(\mathbf{x_t}, \mathbf{x_k}) = k(\mathbf{x_k}, \mathbf{x_t})$ and[3] $\boldsymbol{\alpha}^\top \frac{1}{n} \hat{K} \boldsymbol{\alpha} = \frac{1}{n} \mathbf{a}^\top \mathbf{a} \geq 0$, the matrix $\frac{1}{n} \hat{K}$ is symmetric positive semidefinite and hence its eigenvectors are orthogonal and the corresponding real eigenvalues are non-negative. Let the k positive dominant eigenvalues of $\frac{1}{n} \hat{K}$ be denoted by $\lambda_1 \geq \ldots \geq \lambda_k > 0$ and the corresponding normalized eigenvectors be $\boldsymbol{\alpha}^{\mathbf{1}}, \ldots, \boldsymbol{\alpha}^{\boldsymbol{m}}$. Then the orthogonal matrix of the transformation can be calculated via:

$$Q_{\hat{\phi}} := n^{-1/2} \left[\lambda_1^{-1} \sum_{i=1}^n \boldsymbol{\alpha}^{\mathbf{1}}{}_i \hat{\phi}(\mathbf{x_i}), \ldots, \lambda_k^{-1} \sum_{i=1}^n \boldsymbol{\alpha}^{\boldsymbol{k}}{}_i \hat{\phi}(\mathbf{x_i}) \right], \tag{14}$$

where the factors $n^{-1/2}$ and λ^{-1} are needed to keep the column vectors of $Q_{\hat{\phi}}$ normalized [4].

Transformation of Test Vectors. Let $\mathbf{y} \in \mathcal{X}$ be an arbitrary test vector. New features can be expressed by $\phi(\mathbf{y})^* = W Q_{\hat{\phi}}^\top (\phi(\mathbf{y}) - \boldsymbol{\mu}^\phi)$, where $Q_{\hat{\phi}}$ denotes the matrix we obtained from whitening, while W denotes the orthogonal transformation matrix we got as the output of Kernel-FastICA. Practically speaking, Kernel-FastICA = Kernel-Centering + Kernel-Whitening + iterative section of the original FastICA. Of course, the computation of $\phi(\mathbf{y})^*$ involves only dot products:

$$\phi(\mathbf{y})^* = W n^{-1/2} \left[{\lambda_1}^{-1} \sum_{i=1}^n \boldsymbol{\alpha}^{\mathbf{1}}{}_i c_i, \ldots, {\lambda_k}^{-1} \sum_{i=1}^n \boldsymbol{\alpha}^{\boldsymbol{k}}{}_i c_i \right]^\top, \tag{15}$$

$$c_i = \hat{\phi}(\mathbf{x_i}) \cdot \hat{\phi}(\mathbf{y}) = k(\mathbf{x_i}, \mathbf{y}) - \left(\frac{1}{n} \sum_{j=1}^n \left(k(\mathbf{x_i}, \mathbf{x_j}) + k(\mathbf{x_j}, \mathbf{y})\right) \right) + \frac{1}{n^2} \sum_{t=1}^n \sum_{j=1}^n k(\mathbf{x_t}, \mathbf{x_j}). \tag{16}$$

[3] Here we temporarily disregard the constraint $\mathbf{a} \neq 0$.

[4] If we use the factors $\lambda^{-1/2}$ instead of λ^{-1} in (14), then we obtain the Kernel Principal Component Analysis.

4 Experimental results

In these trials we wanted to see how well independent component analysis and its nonlinear counterpart could reduce the number of features and increase classification performance. Since automatic phoneme classification is of great importance in the computer-assisted training of the speech & hearing handicapped, we chose phoneme classification as an area of investigation.

We developed a program to help with speech training of the hearing impaired, where the intention was to support or replace their diminished auditory feedback with a visual one. In our initial experiments we focussed on the classification of vowels, as the learning of the vowels is the most challenging for the hearing-impaired. The software we designed assumes that the vowels are pronounced in isolation or in the form of two-syllable words, which is a conventional training strategy. Visual feedback is provided on a frame-by-frame basis in the form of flickering letters, their brightness being proportional to the vowels recognizer's output (see fig. 2.).

Corpus. For training and testing purposes we recorded samples from 25 speakers. The speech signals were recorded and stored at a sampling rate of 22050 Hz in 16-bit quality. Each speakers uttered 59 two-syllable Hungarian words of the CVCVC form, where the consonants (C) are mostly unvoiced plosives so as to ease the detection of the vowels (V). The distribution of the 9 vowels (long and short versions were not distinguished) is approximately uniform in the database. In the trials 20 speakers were used for training and 5 for testing.

Feature Sets. The signals were processed in 10 ms frames, from which the log-energies of 24 critical-bands were extracted using FFT and triangular weighting [17]. In our early tests we only utilized the filter-bank log-energies from the most centered frame of the steady-state part of each vowel ("FBLE" set). Then we added the derivatives of these features to model the signal dynamics ("FBLE+Deriv" set). In another experiment we smoothed the feature trajectories so as to remove the effects of short noises and disturbances ("FBLE Smooth" set). In yet another set of features we extended the log-energies with the gravity centers of four frequency bands which approximately corresponds to the possible values of the formants. These gravity centers provide a crude approximation of the formants ("FBLE+Grav" set) [2].

Classifiers. In all the trials with *Artificial Neural Nets* (ANN) [3] the well-known three-layer feed-forward MLP networks were employed with the back-propagation learning rule. The number of hidden neurons was equal to the number of features. In the *Support Vector Machine* (SVM) [21] experiments we always applied the Gaussian RBF kernel function ($k_2,\ r = 10$).

Transformations. In our tests with ICA and Kernel-ICA the eigenvectors belonging to the 16 dominant eigenvalues were selected as basis vectors for the transformed space and the nonlinear function $G(\eta)$ was η^4. In Kernel-ICA the kernel function was as before. Naturally when we applied a certain transformation on the training set before learning, we used the same transformation on the test data during testing.

5 Results and Discussion

Table 1 shows the recognition errors. Here the rows represent the four feature sets, while the columns correspond to the applied transformation and classifier combinations.

On examining the results the first striking point is that although the transformations retained only 16 features, the classifiers could achieve the same or better scores. The reason for this is that ICA determines directions with high non-Gaussianity, which is proven to be a beneficial feature extraction strategy before the classification. As regards the various feature sets, we realized that the gravity center features and smoothing the trajectories both lead to a remarkable improvement in the results, while adding the derivatives in no way increased performance. Most likely, a clever combination of smoothing and taking derivatives (or RASTA filtering) could yield still better results. Another notable observation is that SVM consistently outperformed ANN by several percent. This can mostly be attributed to the fact that the SVM algorithm can deal with overfitting. The latter is a common problem in ANN training.

Finally, with Kernel-ICA, we have to conclude that it is worthwhile continuing doing experiments with this type of nonlinearity. However, the problem of finding the best kernel function for the dot product extension or of choosing the best nonlinearity for the contrast function remains an open one at present.

Table 1. Recognition errors for the vowel classification task. The numbers in parenthesis correspond to the number of features.

	none ANN	none SVM	ICA ANN (16)	ICA SVM (16)	K–ICA ANN (16)	K–ICA SVM (16)
FBLE (24)	26.71%	22.70%	25.65%	23.84%	23.19%	22.20%
FBLE+Deriv (48)	25.82%	24.01%	28.62%	26.81%	24.67%	23.35%
FBLE+Grav (32)	24.01%	22.03%	23.68%	23.35%	20.88%	20.06%
FBLE Smooth (24)	23.68%	21.05%	23.84%	23.84%	22.03%	20.39%

6 Conclusion

In this paper we presented a new nonlinearized version of Independent Component Analysis using a kernel approach. Encouraged by [19] and [8], we could perform further extensions on Kernel Principal Component Analysis (KPCA), since ICA can be viewed as a modified PCA (centering and whitening) and an additional iterative process. But regardless of this we have demonstrated the superiority of Kernel-ICA over its linear counterpart on the phoneme classification task. Unfortunately, feature extraction in kernel feature spaces is currently much slower, than the traditional linear version. Hence in the near future we will focus our effors on working with a sparse data representation scheme that is hoped will speed-up the computations somewhat. This seems to be a good direction to go in.

References

1. Aizerman, M. A., Braverman, E. M. and Rozonoer L. I., Theoretical foundation of the potential function method in pattern recognition learning, *Automat. Remote Cont.*, vol. 25, pp. 821-837, 1964.
2. Albesano, D., De Mori, R., Gemello, R., and Mana, F., A study on the effect of adding new dimensions to trajectories in the acoustic space, *Proc. of EuroSpeech'99,* pp. 1503-1506, 1999.
3. Bishop, C. M., *Neural Networks for Pattern Recognition,* Oxford Univ. Press, 1995.
4. Boser, B. E., Guyon, I. M. and Vapnik V. N, A training algorithm for optimal margin classifier, in *Proc. 5th Annu. ACM Workshop Computat. Learning Theory,* D. Haussler, ed., Pittsburgh, PA, pp. 144-152, 1992.
5. Cristianini, N. and Shawe-Taylor, J. *An Introduction to Support Vector Machines and other kernel-based learning methods* Cambridge University Press, 2000.
6. Comon, P. Independent component analysis, A new concept? *Signal Processing,* 36:287-314, 1994.
7. FastICA Web Page, http:// www.cis.hut.fi/projects/ica/fastica/index.shtml, 2001.
8. Hyvärinen, A. A family of fixed-point algorithms for independent component analysis In *Proceedings of ICASSP*, Munich, Germany, 1997.
9. Hyvärinen, A. New Approximations of Differential Entropy for Independent Component Analysis and Projection Pursuit. In *Advances in Neural Information Processing Systems*, 10:273-279, MIT Press, 1998.
10. Jolliffe, I. J. *Principal Component Analysis,* Springer-Verlag, New York, 1986.
11. Kernel Machines Web Page, http:// www.kernel-machines.org, 2001.
12. Kocsor, A., Tóth, L., Kuba, A. Jr., Kovács, K., Jelasity, M., Gyimóthy, T. and Csirik, J., A Comparative Study of Several Feature Transformation and Learning Methods for Phoneme Classification, *Int. Journal of Speech Technology,* Vol. 3., No. 3/4, pp. 263-276, 2000.
13. Kocsor, A., Kuba, A. Jr. and Tóth, L. Phoneme Classification Using Kernel Principal Component Analysis, *Priodica Polytechnica*, in print, 2001.
14. Kocsor, A., Tóth, L. and Paczolay, D., A Nonlinearized Discriminant Analysis and its Application to Speech Impediment Therapy, *in V. Matousek et al. (eds.): Text, Speech and Dialogue, Proc. of TSD 2001,* Springer Verlag LNAI, in print, 2001.
15. Müller, K.-R., Mika, S., Rätsch, G., Tsuda, K. and Schölkopf, B., An Introduction to Kernel-Based Learning Algorithms, *IEEE Transactions on Neural Networks*, Vol. 12, No. 2, 2001.
16. Mika, S., Rätsch, G., Weston, J., Schölkopf, B. and Müller, K.-R., Fisher Discriminant Analysis with kernels, in *Neural Networks for Signal Processing IX*, Hu, J. Larsen, E. Wilson and S. Douglas, Eds. Piscataway, NJ:IEEE, pp. 41-48, 1999.
17. Rabiner, L. and Juang, B.-H. *Fundamentals of Speech Recognition,* Prentice Hall, 1993.
18. Roth, V. and Steinhage, V., Nonlinear discriminant analysis using kernel functions, In *Advances in Neural Information Processing Systems 12*, S. A. Solla, T. K. Leen and K.-R. Müeller, Eds. Cambridge, MA:MIT Press, pp. 526-532, 2000.
19. Schölkopf, B., Smola, A. J. and Müller, K.-R., *Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput.,* vol. 10, pp. 1299-1319, 1998.
20. Toth, L., Kocsor, A., and Kovács, K., A Discriminative Segmental Speech Model and Its Application to Hungarian Number Recognition, *in Sojka, P. et al.(eds.):Text, Speech and Dialogue, Proceedings of TSD 2000,* Springer Verlag LNAI series, vol. 1902, pp. 307-313, 2000.
21. Vapnik, V. N., *Statistical Learning Theory,* John Wiley & Sons Inc., 1998.