Improved image features by training non-linear diabolo networks

Erik C.D. van der Werf Robert P.W. Duin

Pattern Recognition Group,
Department of Applied Physics, Faculty of Applied Sciences, Delft University of Technology,
P.O. Box 5046, 2600 GA Delft, The Netherlands
erikw@ph.tn.tudelft.nl duin@ph.tn.tudelft.nl

Keywords: Fisher's linear discriminant, neural network, principal components, diabolo network, nonlinear projections, segmentation, auto-associative mapping, feature extraction.

Abstract

This paper discusses a trainable system to extract features for image segmentation based on non-linear mapping of local features.

Supervised training methods are presented, for artificial neural diabolo networks, which produce a mapping comparable to Fisher's linear discriminant mapping. This mapping can be used to decrease dimensionality whilst preserving class separability.

It is shown that the non-linear feature extraction performed in diabolo networks can increase class separability, compared to linear mapping methods, thus resulting in improved image segmentation.

1 Introduction

To handle a selective image database search, low-level properties in images can be calculated in a local neighbourhood. Algorithms that calculate such low-level properties are called filters. Many special purpose applications have been developed which apply problem specific filters. For well-defined tasks a small number of these filters often is sufficient to characterise different image regions. However if the important features are not previously known, either because there is no expert on the subject, because the problem is too complex, or because examples of the objects are not given in advance, all one can do is apply more filters and search for a useful combination.

In general a filter performs a measurement on the local neighbourhood around each pixel. In pattern recognition such measurements are called features. If more filters are applied to an image each pixel has its own feature-vector, built from all measurements, which can be used to classify the pixels, finally resulting in image segmentation. In most cases however, these initial feature-vectors will not be efficient. They are not efficient since features can be redundant, correlated or non-linearly related and the computational cost of calculations grows with the number of features. Another problem is the fact that distances loose meaning in high dimensional space. To obtain more efficient feature-vectors it is necessary to reduce the dimensionality of the feature-space. Methods that do this are feature-selection and feature-extraction methods. In this paper we focus at feature-extraction. An interesting approach to feature selection, which might also be applied to our neural networks, can be found in [2].

In pattern recognition there is a wide range of methods for feature extraction such as principal component analysis, statistical discriminant analysis, independent component analysis and Kohonen mapping. This paper describes and analyses a feature extraction method based on the non-linear mapping of original features onto a lower-dimensional subspace with feed-forward neural networks.

The rest of this paper is organised as follows: first in section 2 some feature extraction methods that can be implemented in feed forward neural networks are described. In section 3 a training algorithm for non-linear diabolo networks is given. In section 4 some experiments are shown. Finally, in the last section, some conclusions are drawn.

2 Mapping methods

A process that has as input a n-dimensional feature-vector **x** and as output a m-dimensional feature-vector **y**, m<n, is called a feature extraction method or mapping method. The goal of projecting the

original n-dimensional feature-vector onto an m-dimensional subspace is to get a more efficient combination of the original features.

Mapping methods can either be linear or non-linear and supervised or unsupervised. The difference between supervised and unsupervised methods is whether or not class information is used.

2.1 Unsupervised mapping methods

The most widely used linear mapping is the Principal Component Analysis (PCA) also known as the Karhunen-Loève transform. This unsupervised mapping method is a projection method that assumes the best mapping to preserve the maximum amount of variance. It can be shown [1] that the optimal linear solution for representing the n-dimensional vectors x in an m-dimensional space, m<n, is to project x onto the surface spanned by the m largest eigenvectors of the covariance matrix.

It is easily shown that a 3-layer feed-forward neural network with n inputs, m neurons in the second layer and n neurons in the third layer, with all linear activation functions, is able to perform a PCA mapping between the first and the second layer. Such neural networks are often called auto-associative or diabolo networks. A schematic overview of a linear diabolo network is shown in Figure 1.

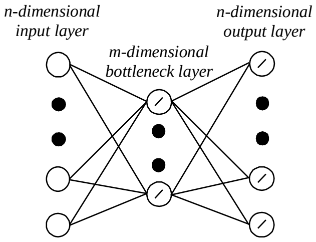

Figure 1, linear diabolo network

To learn to approximate a PCA mapping between the input and bottleneck layer and a reconstruction between the bottleneck and output layer, the output of the network is trained to approximate the input. So the inputs are also the targets. After the network is trained it is split and the first half, between the input and bottleneck layer, is used for extracting the m new features. The main difference with normal PCA is that, in general, the extracted features will be non-orthogonal rotated versions of the features extracted by normal PCA. Furthermore it turns out that for practical applications the normal PCA, which is non-iterative, is trained much faster.

An extension of the PCA network is the Non-Linear Principal Component Analysis (NLPCA) network. The difference with linear PCA networks is that this network has extra hidden layers between in- and output and the bottleneck-layer. The neurons in these extra hidden layers have non-linear activation functions, which allow the network to find non-linear subspaces.

The smallest NLPCA network with non-linear compression and reconstruction, shown in Figure 2, uses five layers. The first and the fifth layer are the n inputs and outputs. The third layer has m neurons, usually with linear activation functions. The neurons in the second and fourth layer perform the non-linear transformation. The number of neurons in these layers depends on the amount of non-linearity in the compression and reconstruction of the data. After the network is trained it is split and the first half, between the input and bottleneck layer, is used for extracting the m new features.

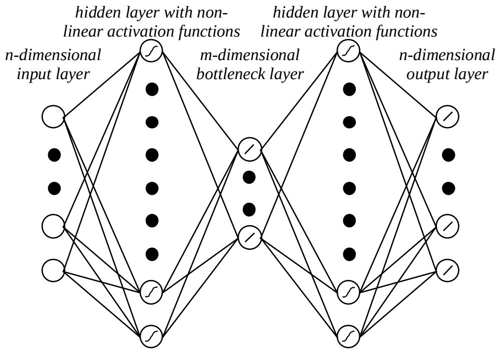

Figure 2, Non-linear diabolo network

2.2 Supervised mapping methods

When we have two or more classes, feature extraction is equivalent to the choice of the mapping, which is most effective for showing class separability. In statistical discriminant analysis, within-class, between-class and mixture scatter matrices are used to formulate criteria of class separability. The within-class scatter matrix shows the scatter of samples $\mathbf{x}_i$ around their class expected vector μ_i, and is expressed by

$$S_w = \sum_{i=1}^{k} P_i E[(x_i - \mu_i)(x_i - \mu_i)^T] \ , \qquad (1)$$

a between-class scatter matrix can be defined as

$$S_b = \sum_{i=1}^{k} P_i (\mu_i - \mu)(\mu_i - \mu)^T \ , \qquad (2)$$

the mixture scatter matrix is the covariance matrix of all samples regardless of their class assignments, and is defined by

$$S_m = E[(x-\mu)(x-\mu)^T] = S_w + S_b \,. \qquad (3)$$

P_i is the a priori probability of class i, k is the number of classes and μ is the mean of all vectors regardless of class assignment.

In order to formulate criteria for class separability, we have to derive a number from these matrices. There are many ways to do this. In general the number should be larger when the between-class scatter is larger or the within-class scatter is smaller. A criterion commonly used for linear mapping methods is

$$J = tr\left(\frac{S_1}{S_2}\right), \qquad (4)$$

in which S_1 and S_2 are one of S_b , S_w or S_m. In our experiments we used S_b for S_1 and S_w for S_2.

It can be shown [1] that the optimal linear solution with respect to J for representing the n-dimensional feature-vectors **x** in an m-dimensional subspace, m<n, is to project **x** onto the surface spanned by the m largest eigenvectors of $S_2^{-1}S_1$. This projection is known as Fisher's linear discriminant mapping (FLD).

Although FLD mapping has proven to be a useful tool in pattern recognition, it is limited to linear projections.

Since diabolo networks can learn linear and non-linear PCA mapping by unsupervised training of the output to approximate the input [3], we can ask the question if a similar training method, with targets presented at the output of the network, could be applied to learn FLD mapping.

The first question in training a diabolo network to approximate FLD mapping is what targets should be used. For the PCA network we chose the input vectors as targets. While this approach ensures us that the global structure is preserved, it does not necessarily improve class separability. Since we choose to train our diabolo networks with targets presented at the output of the network, we cannot directly optimise class separability in the bottleneck layer. We can however try to optimise class separability at the output of the network. If the reconstructed feature-space is well separable the same should hold for the extracted features in the bottleneck layer, although it should be noted that the job of actually separating the classes in the bottleneck layer might be harder due to the non-linear transformations.

To enhance class separability we would like a contraction of each class. Ideally each class would be projected in one unique point. Therefor to train a diabolo network to find a good separable mapping we use one unique point per class as the target.

A criterion that should be optimised for the targets is the preservation of the global structure of the reconstructed feature-space. To do this we can define a scatter measure, similar to (1), showing the scatter of samples $\mathbf{x}_\mathrm{i}$ around their target vector $\mathbf{t}_\mathrm{i}$ as

$$S_{tw} = \sum_{i=1}^{k} P_i E[(x_i - t_i)(x_i - t_i)^T] \,, \qquad (5)$$

in which P_i is the a priori probability of class i and k is the number of classes.

It can easily be shown that the trace of S_{tw} (5), which is the mean-square distance to targets, is minimised by choosing the class means as targets. In most applications these targets perform well. In some cases however they can create a problem since the distances between targets of overlapping classes or classes having strange distributions could become small, thus reducing separability.

To overcome problems for most class distributions a second scatter measure can be devised which shows the scatter of the target vectors $\mathbf{t}_\mathrm{i}$ around the expected vector, regardless of class assignment, μ as

$$S_{tb} = \sum_{i=1}^{k} P_i E[(t_i - \mu)(t_i - \mu)^T] \,, \qquad (6)$$

in which P_i is the a priori probability of class i and k is the number of classes.

With these two scatter matrices targets can be calculated by iterative maximisation of J, using S_{tb} for S_1 and S_{tw} for S_2. Our choice for J is motivated by the fact that this criterion is used for FLD mapping and therefor might be useful for comparing NLFLD mapping to FLD mapping. In general however there is a much wider variety of clustering techniques, optimising other criteria, that could be applied to calculate targets, this however is beyond the scope of this paper.

One important difference, that should be kept in mind when comparing normal FLD mapping to mapping with a diabolo network, is that the diabolo network is trained to minimise distances to class assigned targets. This means that for a diabolo network there will be a trade-off between minimising within-scatter and restoring the between-scatter associated with the target positions. For normal FLD mapping the trade-off is between minimising within-scatter and maximising between-scatter.

3 Training the network

Training a non-linear diabolo network generally takes quite a long time to reach an optimal performance. To improve training speed we initialised our networks with an approximation of the best linear mapping. This is done in five steps:

1. First a linear mapping and reconstruction is calculated which results in a linear diabolo network with one hidden bottleneck layer with m neurons.
2. For obtaining a five-layer diabolo network two extra layers are added between the hidden layer and the output layer. Initially these extra layers use m neurons, with linear activation functions. The weights and biases of the new layers are set to perform a unity mapping, thus ensuring that the output of the network remains the same. The middle of the three m-dimensional layers becomes the new bottleneck layer.
3. The m neurons in the layers around the bottleneck layer are copied a number of times and the connection weights are divided by the number of copies so that the inputs to the next layers remain the same.
4. The linear activation functions, of the neurons in the layers around the bottleneck layer, are replaced with non-linear hyperbolic tangent sigmoid activation functions. The weights and biases are adjusted so that the data is approximately in the linear part of the activation function.
5. Finally to avoid symmetry problems in training, due to identical neurons in the same layer, noise is added to all weights and biases.

The now obtained diabolo network is trained further with standard back-propagation training algorithms.

4 The Experiments

Several experiments were performed to investigate the power of non-linear mapping methods for image segmentation. Some results are presented in the following.

In our experiments a 256x256 image of Lena was used. The image is shown in Figure 3. From this image 4 Intensity, 9 DCT, 8 Gabor, 4 Wavelet and 12 Colour features were calculated in a local 9x9 window around each pixel. We manually selected regions of the classes skin, the hat, the boa, hair and the background. From these regions 200 samples per class were selected for learning and another 200 samples per class were selected for testing. These samples were selected randomly under the constraint that the local window around pixels selected for learning did not overlap the window around pixels selected for testing.

With samples selected for learning, several different 5-layer diabolo networks were trained. These networks were different in the sense that we tried different numbers of neurons for all hidden layers.

For training the networks we used gradient descent with momentum and adaptive learning back-propagation. From the training examples 90 percent was used for training and 10 percent was used for validation. We trained for 10000 epochs until the performance on the validation set decreased over 100 epochs. If the training had not reached 10000 epochs or performance had not increased, training was restarted for at most five times with other training and validation sets randomly chosen from the 1000 learn vectors.

After comparing the networks we concluded that the best performance would be obtained using a 4-dimensional bottleneck. For the linear case this is obvious since five class-means span a 4-dimensional subspace, for non-linear mapping this relation is less obvious since a lower-dimensional subspace could also curve through all four class-means. However, using only two neurons in the bottleneck layer and 12 neurons in both non-linear hidden layers seemed more instructive for comparing the different mapping methods. Furthermore, it turned out that for this image the best results did not decrease much by going from a 4-dimensional subspace to a 2-dimensional subspace. Another advantage of using a 2-dimensional mapping is that it is well suited for visual inspection of the feature-space.

Figure 3, Lena

After selecting a new learn set and test set, following the same procedure as before, four new mappings were trained. All networks were trained to map the data onto a 2-dimensional subspace, using 12 neurons in the hidden layers for compression and reconstruction. After training each network was split and the first half was used to extract the two new features. These features, still all from pixels selected for learning, were used to train a Mahalanobis classifier and a nearest mean classifier. These classifiers were then tested on all features extracted from the test-set. In Table 1 the percentages of errors are shown for the different mapping methods. The classification error of both classifiers applied to the original 37-dimensional feature-vectors was 34.1 % for the nearest mean classifier and 10.7 % for the Mahalanobis classifier with optimal regularisation.

Classifier	PCA	NLPCA	FLD	NLFLD
Nearest mean	44.2 %	38.1 %	15.9 %	8.8 %
Mahalanobis	36.7 %	32.2 %	14.7 %	8.5 %

Table 1, Classification errors after mapping

To see how well the diabolo networks had optimised class-separability, the performance of the different mapping methods was calculated with criterion *J* (4). The results are shown in Table 2. Scatter-plots of the associated feature-spaces are shown in Figure 4

	PCA	NLPCA	FLD	NLFLD
J	1.89	1.93	11.85	12.65

Table 2, Class separability

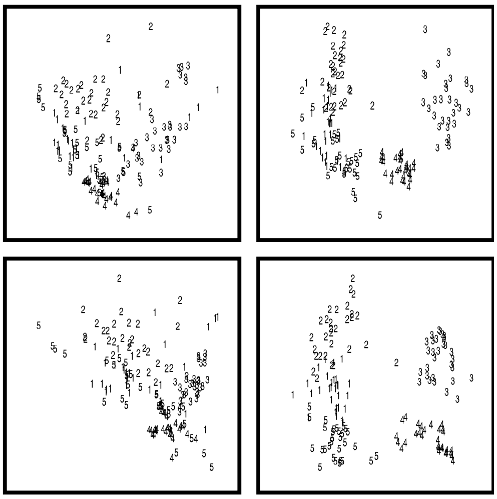

Figure 4, Extracted 2d-features of 150 test-samples. Left top PCA, right top FLD, left bottom NLPCA, right bottom NLFLD.

To illustrate some differences between the mapping methods, we used the same Mahalanobis classifiers to segment the whole image. The results are shown in Figure 5. The reader should keep in mind that the question of what image looks better is subjective, that the result is influenced by our choice for equal probabilities and less than 50% of the images has actually been used for training the classifier. The different feature extraction methods should therefor mainly be judged on their performance on the test set, and not on the segmented images.

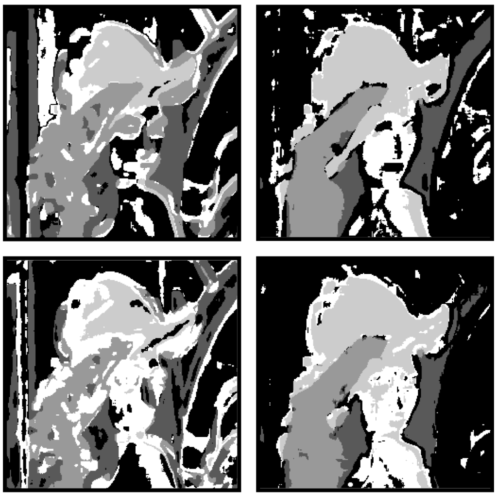

Figure 5, Segmented images of Lena. Left top PCA, right top FLD, left bottom NLPCA, right bottom NLFLD.

5 Conclusion and discussion

In this paper a new supervised method for training non-linear diabolo networks was presented which can improve feature extraction compared to linear PCA and FLD mappings.

We've shown that non-linear diabolo networks can effectively learn to reduce dimensionality and at the same time increase class separability. In general the reduction of dimensionality performed by non-linear mappings can aid in improved speed and performance for all distance based classifiers. Combined with feature-selection methods the diabolo network is a valuable tool for fast selective search through high dimensional data.

In our experiments we've shown that the performance of image segmentation algorithms, that apply filter banks, can be improved with non-linear diabolo networks. Class separability has been increased for the NLFLD mapping. The NLPCA mapping also shows a slight increase. The result, however, is not as much as we might have hoped. Better results may be obtained by repeating the experiment several times, with larger learn-regions, and averaging a leave-one-out estimate obtained from a smaller test-region.

A remaining problem is the choice for the number of neurons and hidden layers that should be used. In general the number of freedoms should be as low as possible, since each extra neuron increases the computational complexity and the chance of over training. However aside from some rough estimates for the intrinsic dimensionality most parameters still have to be tuned on a trial and error basis.

In this paper we presented a mapping that performs both extraction and reconstruction of the original feature-space, in many applications the reconstruction of the original feature-space will not be used and training methods that do not need calculation of the reconstruction of the feature-space might be favoured over diabolo networks. An approach might be to calculate the derivative of *J* (4), for data projected to an m-dimensional output-layer, in our case the bottleneck layer, and use it to train the network with back-propagation. A problem however is that the direct minimisation of J is computationally much more intensive than just minimising distances, furthermore the criterion itself is questionable.

In our research we found some discomforting properties of criterion *J* (4), which is used for optimising Fisher's Linear Discriminant mapping. For the case of simple class distributions and a small number of classes this criterion performs well. If the number of classes gets high, the extracted feature-space may not be optimal for classification purposes. Since *J* is only invariant for translation and rotation of the feature-space, non-linear transformations may give results that are partially irrelevant with respect to how well classes are separable. Its use is therefore questionable. In our approach we only used it to calculate targets and possible problems may therefore be overcome by applying better clustering techniques. However, non-linear mapping methods that optimise *J* directly should be treated with caution.

Acknowledgements

We would like to thank Kieron Messer from the Centre for Vision Speech and Signal Processing, at the University of Surrey, for the filters that were used in our experiments.

References

[1] K. Fukunaga: Introduction statistical pattern recognition, Academic Press Inc, 1990

[2] K. Messer, J.Kittler and M. Kraaijveld: Selecting features for neural networks to aid an Iconic search through an image database, IEE 6th International Conference in Image Processing and its Applications pages 428-432, University of Surrey, 1997

[3] E. Oja: Data Compression, Feature Extraction, and Autoassociation in Feedforward Neural Networks, Elsevier Science Publishers B.V., 1991.