Dimensionality Reduction Using a Novel Neural Network Based Feature Extraction Method

S. J. Perantonis and V. Virvilis

Institute of Informatics and Telecommunications,
National Center for Scientific Research "Demokritos", Athens, Greece

1. Abstract

A novel neural network based method for feature extraction is proposed. The method achieves dimensionality reduction of input vectors used for supervised learning problems. Combinations of the original features are formed that maximize the sensitivity of the network's outputs with respect to variations of its inputs. The method exhibits some similarity to Principal Component Analysis, but also takes into account supervised character of the learning task. It is applied to classification problems leading to efficient dimensionality reduction and increased generalization ability.

2. Introduction

Methods for dimensionality reduction concentrate either on *selecting* from the original set of features a smaller subset of salient features, or on *combining* the original features in such a way as to *extract* a small number of salient features. Application of such methods to data analysis or pattern recognition problems has distinct advantages in terms of generalization properties and processing speed. In this respect, feature extraction methods are probably preferable to feature selection techniques because they usually result in feature vectors of lower dimensionality. Many classification or function approximation algorithms exhibit greatly improved speed of convergence and generalization properties when called upon to operate using training sets of reduced dimensionality.

Ruck [1] has developed a feature selection method based on the application of multilayer feedforward networks (MFNN). According to this method, a saliency metric is defined that depends on the sensitivity of the trained network outputs with respect to its inputs. The MFNN is preliminarily trained using all available features whose saliencies are subsequently determined. The most salient features are then selected for further processing. An advantage of the method is that it forms salient features taking into account information about the classification or function approximation problem itself. Indeed, the feature selection process is closely related to a pretraining procedure whereby the desired targets are utilized. A drawback of this method is that it just selects features from the original set of available features, but does not consider further dimensionality reduction by forming salient combinations of the original features.

The objective of this paper is to present an extension of Ruck's method that still takes into account the supervised character of the learning task, but also is capable of performing feature extraction by proposing salient combinations of the original features. The proposed salient features are linear combinations of the original features. These combinations are selected to maximize the sensitivity of the network outputs to small variations of the inputs. The method bears similarities to Principal Component Analysis (PCA) [2][3][4], with the important difference that the eigenvalue problem is formulated using a matrix depending on the weights of the pretrained neural network instead of the covariance matrix of the inputs.

The method is tested in classification problems. Once features are extracted, each problem is solved using two different learning paradigms (MFNNs and nearest neighbor classifiers). Results are compared to those obtained by other feature selection/extraction methods (Ruck's method, PCA, t-test method). Two types of benefits arise from our simulation results. Firstly, an increase in generalization ability is observed. Secondly, in comparison with other methods, the dimension of the extracted feature vector is usually greatly reduced. This can offer an increase in processing speed, particularly in conjunction with supervised classification methods (e.g. k-nn) whereby the effectiveness of

fast method variants is heavily dependent on the dimensionality of the input vector.

3. Proposed Method

Consider a MFNN with one layer of input, M layers of hidden and one layer of output units. The units in each layer receive input from all units in the previous layer. Inputs to the first layer of the MFNN are denoted by $x_i, i = 1, \ldots, N$ where N is the total number of features the network is called upon to process. Output units are denoted by $O_i^{(m)}$, where the superscript (m) labels a layer within the structure of the neural network ($m = 1, 2, \ldots, M$ for the hidden layers, $m = M + 1$ for the output layer), and i labels a unit within a layer. The synaptic weights are denoted by $w_{i_{m-1}i_m}^{(m)}$, where m, i_m denote respectively the layer and the unit toward which the synapse is directed and i_{m-1} denotes the unit in the previous layer from which the synapse emanates. Biases will be treated as weights emanating from units with constant, pattern-independent output equal to one. The logistic function $f(s) = 1/(1 + \exp(-s))$ is used as the activation function of hidden and output units.

Consider the vector space $\mathcal{V}$ spanned by all possible feature vectors $\boldsymbol{x}$. Given a particular direction defined by a unit vector $\hat{\boldsymbol{u}}$ belonging to $\mathcal{V}$, a saliency metric will be introduced which is designed to express the sensitivity of the network's output to small perturbations of the input vectors along this direction. Given a vector $\boldsymbol{x}$, let us denote by $x_{\hat{u}}$ its projection along the direction $\hat{u}$, i.e. $x_{\hat{u}} = \boldsymbol{x} \cdot \hat{\boldsymbol{u}}$. Then the saliency along the direction $\hat{\boldsymbol{u}}$ is defined by:

$$S_{\hat{u}} = \sum_{\{\boldsymbol{x}\}} \sum_i \left(\frac{\partial O_i^{(M+1)}}{\partial x_{\hat{u}}} \right)^2 \tag{1}$$

The first sum in the above expression is formed using different randomly chosen input vectors. We seek to find those directions $\hat{\boldsymbol{u}}$, for which the corresponding saliency $S_{\hat{u}}$ is extremal, subject to the constraint $\hat{\boldsymbol{u}} \cdot \hat{\boldsymbol{u}} = 1$. We shall show that this problem reduces to the eigenvalue problem of a real symmetric matrix. Indeed, by employing the well known property of the directional derivative:

$$\frac{\partial O_i^{(M+1)}}{\partial x_{\hat{u}}} = \sum_k \hat{u}_k \frac{\partial O_i^{(M+1)}}{\partial x_k}, \tag{2}$$

we readily obtain the following expression for the saliency S_u:

$$S_{\hat{u}} = \sum_{j,k} R_{jk} \hat{u_j} \hat{u_k} \tag{3}$$

where

$$R_{jk} = \sum_{\{\boldsymbol{x}\}} \sum_i \frac{\partial O_i^{(M+1)}}{\partial x_j} \frac{\partial O_i^{(M+1)}}{\partial x_k} \tag{4}$$

is a symmetric matrix whose elements are readily calculated using the formula:

$$\frac{\partial O_{i_{M+1}}^{(M+1)}}{\partial x_{i_0}} = \sum_{i_1, i_2, \ldots, i_M} \prod_{m=1}^{M+1} O_{i_m}^{(m)} \left(1 - O_{i_m}^{(m)}\right) w_{i_{m-1} i_m}^{(m)} \tag{5}$$

It is now required to maximize expression (3) with respect to $\hat{u_k}$, subject to the constraint $\sum_k \hat{u_k} \hat{u_k} = 1$. On introducing a Lagrange multiplier μ to take account of the constraint, we form the expression

$$S'_{\hat{u}} = \sum_{j,k} R_{jk} \hat{u_j} \hat{u_k} + \mu (1 - \sum_k \hat{u_k} \hat{u_k}). \tag{6}$$

Constrained extrema of $S_{\hat{u}}$ occur when $\partial S'_{\hat{u}} / \partial \hat{u_j} = 0$, so that

$$\sum_k R_{jk} \hat{u_k} = \mu \hat{u_j}. \tag{7}$$

It follows that the constrained extrema occur when $\hat{\boldsymbol{u}}$ is an eigenvector of $\boldsymbol{R}$. Substituting (6) into (3) and taking account of the constraint, we readily conclude that $S_{\hat{u}} = \mu$, so that maximum saliency is equal to the maximum eigenvalue of $\boldsymbol{R}$ and is found when $\hat{\boldsymbol{u}}$ is the eigenvector of $\boldsymbol{R}$ corresponding to its maximum eigenvalue.

As a result of the above discussion, the following feature extraction method is proposed: The MFNN is "pretrained" using all available features, preferably a number of times using different initial weights. Once pretraining is completed, elements of the matrix $\boldsymbol{R}$ are computed using (4) and (5). If more than one training sessions are involved, the first sum in equation (4) includes information from all training sessions. Given a saliency threshold S_N, let there exist K eigenvalues of $\boldsymbol{R}$ larger than S_N. The eigenvectors $\hat{\boldsymbol{u}}^r, \quad r = 1, \ldots K$ of $\boldsymbol{R}$ corresponding to these eigenvalues are evaluated and the K salient features extracted by our method are given by $\boldsymbol{x} \cdot \hat{\boldsymbol{u}}^r, \quad r = 1, \ldots K$. The newly computed salient features can then be used to train either a MFNN or any other supervised learning paradigm.

4. Relation to Other Methods

The feature selection method of Ruck and collaborators [1] is based on pretraining a MFNN to solve a specific learning task and arranging input features in descending order of saliency using the following saliency metric:

$$S_j = \sum_{\{\boldsymbol{x}\}} \sum_i \left\| \frac{\partial O_i^{(M+1)}}{\partial x_j} \right\|_1 \tag{8}$$

where the first sum denotes inclusion of information from all pretraining sessions and input patterns. Clearly, Ruck's metric is similar to the metric proposed in this work, but involves only the partial derivative of the outputs with respect to each input and not to input combinations. Hence, only directional derivatives with respect to the basis vector directions of space $\mathcal{V}$ are involved. Also, the Ruck metric employs the 1-norm of the output vector derivative with respect to input features, whereas our metric employs the 2-norm. However, the order of the norm does not seem to be important for selecting salient features. Indeed, a similar saliency metric proposed by Tarr [5] is more closely related to the 2-norm of this derivative.

The relation of our method to PCA is also evident. PCA amounts to solving a constrained optimization problem similar to the one introduced by the Lagrangian of (6). However, in PCA, the matrix $\boldsymbol{R}$ is the covariance matrix of the input patterns. Clearly, our method retains the advantage of forming salient linear combinations of the original features, but also incorporates in the saliency metric information about the desired targets of the supervised learning problem.

5. Simulations

We use synthetic and real world examples to evaluate our method. An interesting synthetic example is the *rotated XOR (R-XOR)* problem. Consider P two-dimensional vectors (x_1, x_2) uniformly sampled from the square defined by $-1 < x_1 < 1$ and $-1 < x_2 < 1$. In the usual XOR problem, there are two classes. Vectors whose components obey $x_1 x_2 > 0$ belong to Class 1, while vectors obeying $x_1 x_2 < 0$ belong to Class 2. We add six distractor features (x_3, x_4, x_5, x_6, x_7 and x_8), all randomly sampled between -1 and 1, and rotate each vector in the eight dimensional space defined by the $x_i, i = 1, \ldots, 8$ by an arbitrary rotation operator $\boldsymbol{A}$. The "rotated XOR problem" is defined as follows: A rotated vector $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ belongs to Class 1, if $x_1 x_2 > 0$ and to Class 2 if $x_1 x_2 < 0$. Note that in the rotated XOR problem all features $y_i, i = 1, \ldots, 8$ play a role in the final classification result, but only two linear combinations of these features are salient. A sample of 200 vectors was used to implement the rotated XOR problem. We also give results concerning supervised learning examples from the University of California-Irvine machine learning repository [6], namely the *BUPA Liver Disorders* set and the *Ionosphere* set [7]. Both tasks are classification problems with two classes.

Apart from the method proposed in this work, we also give results from the application of other feature selection or feature extraction methods, namely Ruck's method, Tarr's method and PCA. The salient features determined by each feature extraction/selection method are given as inputs to two types of classifier, namely a MFNN and a nearest neighbor classifier.

	R-XOR	BUPA	IONO
Proposed	91.3 (3)	72.4 (2)	95.4 (4)
Tarr	84.6 (7)	69.4 (5)	92.8 (8)
Ruck	86.2 (7)	70.4 (4)	93.5 (12)
PCA	86.1 (7)	69.6 (5)	95.4 (18)
Original	87.3 (8)	70.5 (6)	94.5 (33)

Table 1: Generalization ability (average classification accuracy in the test sets) achieved using a feedforward network to which the results of various feature extraction/selection methods are given as input. The number of salient features used to achieve each result are given in parentheses.

To assess generalization ability, each dataset was partitioned into a training set consisting of 75 % of the available input vectors and a test set consisting of the remaining 25% of the data. Five different partitions were chosen at random. Generalization ability results are given as averages over the 5 test sets. All pretraining sessions (where relevant) and final training sessions for MFNN were performed using an efficient variation of the backpropagation algorithm based on the adaptive use of momentum acceleration [8]. The values $\delta P = 0.3$ and $\xi = 0.5$ were used for the gain δP and the momentum regulator ξ respectively for all problems. For all benchmarks, training was carried on for at most 200 epochs or until the mean squared error dropped below the value $2 \cdot 10^{-3}$. Networks with one hidden layer were used for all problems. For the rotated XOR problem, the hidden layer had 4 units. For all other problems 10 hidden units were used. In order to compute saliencies for the proposed method, 5 pretraining sessions with different randomly chosen initial weights were performed for each of the 5 training sets.

The results of our simulations are summarized in Tables 1 and 2. In Table 1, generalization ability results are presented for MFNNs trained using the salient features found by each feature selection/extraction method. The number of features used in conjunction with each method are also given in parentheses. The quoted number of features is that for which maximum generalization ability is obtained (subject to the constraint that some feature selection

	R-XOR	BUPA	IONO
Proposed	92.5	70.9	94.3
Tarr	67.5	59.3	90.3
Ruck	67.5	65.1	91.4
PCA	70.0	63.4	93.7
Original	65.0	61.6	93.1

Table 2: Generalization ability (average classification accuracy in the test sets) achieved using a nearest neighbor classifier to which the results of various feature extraction/selection methods are given as input. The same salient features are used as in the previous table.

is performed, i.e. at least one of the original features is eliminated). In Table 2, generalization ability results are presented for nearest neighbor classifiers using the same salient features.

In the three benchmarks, an increase in generalization ability is observed with respect to the original set of features. This is true for both the MFNN and nearest neighbor classifiers. Naturally, best results were obtained in the synthetic rotated XOR problem, where it is known that the salient features are indeed linear combinations of a subset of the original features. Moreover, as is evident from Table 1, our method has succeeded in extracting a relatively low number of significant features in all three benchmarks. This characteristic is clearly not shared by any of the other methods. This characteristic of the method may prove important for processing speed efficiency in applications where large training sets and original feature space dimensionalities are involved. In such applications, even in the testing phase, some types of classifiers become prohibitively slow. A notable example is the nearest neighbor classifier. Most fast implementations of this classifier work well and provide acceleration only when the feature space dimensionality is low (usually less than 15 dimensions) [9]. We are currently working on the application of the method to a large scale OCR problem. Preliminary results show that significant acceleration (one to two orders of magnitude) is observed using the Nene-Nayar variant of the nearest neighbor classifier [10] applied to the salient features determined by the proposed feature extraction method.

6. Conclusion

In this paper, a new method was proposed for feature extraction and dimensionality reduction. The method is based on pretraining an MFNN and extracting salient linear combinations of the original features depending on the sensitivity of outputs to variations of the original features. The method was tested in conjunction with MFNN and nearest neighbor classifiers in synthetic and real world benchmarks and was shown to lead to significant dimensionality reduction and increased generalization ability. The effects of the application of the method on processing speed were also discussed.

References

[1] D. W. Ruck, S. K. Rogers and M. Kabrisky, "Feature selection using a multilayer perceptron", *Neural Network Comput.*, Vol. 2, pp. 40-48, 1990.

[2] K. Karhunen, "Ueber lineare Methoden in der Wahrscheinlichkeitsrechnung", *Annales Academiae Scientiarum Fennicae*, Series A1: Mathematica-Physica, Vol. 37, pp. 3-79, 1947.

[3] M. Loéve, *Probability Theory, 3rd ed.*, New York: Van Nostrand, 1963.

[4] E. Oja, "A simplified neuron model as a principal component analyser", *Journal of Mathematical Biology*, Vol. 15, pp. 267-273, 1982.

[5] G. Tarr, Multilayered Feedforward Networks for Image Segmentation, Ph.D. Thesis, Air Force Institute of Technology, 1991.

[6] P. M. Murphy and D. W. Aha, "UCI repository of machine learning databases, Machine readable data repository, Irvine, CA". University of California, Department of Information and Computer Science, 1992.

[7] V. G. Sigillito, S. P. Wing, L. V. Hutton and K. B. Baker, "Classification of radar returns from the ionosphere using neural networks", *Johns Hopkins APL Tech. Digest*, Vol. 10, pp. 262-266, 1989.

[8] S. J. Perantonis and D. A. Karras, "An efficient constrained learning algorithm with momentum acceleration", *Neural Networks*, Vol. 8(2), pp. 237-239, 1995.

[9] J. McNames, "A nearest trajectory strategy for time series prediction", *Proc. International Workshop on Advanced Black-Box Techniques for Nonlinear Modeling*, KU Leuven, Belgium, pp. 112-128, 1998.

[10] S. A. Nene and S. K. Nayar, "A simple algorithm for nearest neighbor search in high dimensions", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, pp. 989-1003, 1997.