ARTIFICIAL INTELLIGENCE AND SOFT COMPUTING
August 28–30, 2006, Palma de Mallorca, Spain
ISBN Hardcopy: 0-88986-610-4 / CD: 0-88986-612-0

GLOBAL-LOCAL LEARNING STRATEGIES IN PROBABILISTIC PRINCIPAL COMPONENTS ANALYSIS

Ezequiel López-Rubio, Juan Miguel Ortiz-de-Lazcano-Lobato, Domingo López-Rodríguez, Enrique Mérida-Casermeiro, María del Carmen Vargas-González
School of Computer Engineering, University of Málaga,
Campus de Teatinos, s/n, 29071 Málaga
Spain
{ezeqlr, jmortiz}@lcc.uma.es, {dlopez, merida}@ctima.uma.es

ABSTRACT
We present a neural model which extends classical competitive learning by performing a Probabilistic Principal Components Analysis at each neuron. In the learning process is utilized a competition rule which try to get the better representation of the dataset while maintaining the homogeneity of the formed clusters. The model also has the ability to learn the number of basis vectors required to represent the principal directions of each cluster, so it overcomes a drawback of most local PCA models, where the dimensionality of a cluster must be fixed a priori.

KEY WORDS
Neural networks, local PCA, competitive learning

1. Introduction

The *Principal Components Analysis* is a multispectral data analysis technique, which is aimed to obtain the principal directions of the data, i.e., the maximum variance directions (see [**1**], [**2**]). Hence, if we have a D-dimensional input space, the PCA computes the K principal directions, where $K<D$. This allows important dimensionality reductions by selecting $K<<D$. It has been proved that PCA is an optimal linear technique for dimensionality reduction, in the mean sense (see [**1**]).
The original method, sometimes called *Karhunen-Loève (KL) transform* or *global PCA*, considers the entire input distribution as a whole and this limits its applicability. A number of *local PCA methods* have been proposed to partition the distribution into meaningful clusters (see [**3**], [**4**], [**5**]). These methods have been widely used in the context of transform coding to compress multispectral and multilayer images (see [**6**], [**7**]). Furthermore, they are used in visualization of high-dimensional data, which requires mapping to a lower dimension (typically, $K\leq3$). Alternatively nonlinear extensions of PCA, [**8**], have also been used for this task but they involve much more complexity. Some of the previous techniques as [**5**] derive PCA from the perspective of density estimation and offer a number of important advantages. Nevertheless, all those local PCA procedures do not address the problem of selecting the correct number of basis vectors K. This drawback has been studied in the context of global PCA by several authors (for example, [**9**] and [**10**]).
The model we have developed combines vector quantization and principal component analysis to obtain a new neural model, which adjusts the exact dimension required for a cluster, in order to improve the representation of the samples it encompasses without human help. The competitive learning algorithm has been designed to partition the input space into a set of regions, which satisfies a compromise between the homogeneity of the clusters and the faithful representation capacity of the input data. Then, a probabilistic Gaussian model captures the main statistical information, of each cluster, i.e. the mean and the variance of the principal directions. In every learning epoch we ensure that a minimum percentage of the variance is retained, which usually involves a change in the number of basis vectors that must be conserved for each cluster.
This paper is structured as follows. Section 2 reviews some probabilistic PCA concepts. We describe our model in Section 3. Section 4 shows the experimental results that we have obtained with a real data set, and conclusions are discussed in Section 5.

2. Probabilistic PCA

A latent variable model seeks to relate a D-dimensional observed data vector $\mathbf{t}$ to a corresponding Q-dimensional vector of latent variables $\mathbf{x}$ whose values are unknown. Commonly, a linear function is used to map from latent subspace to data space:

$$\mathbf{t} = \mathbf{W}\mathbf{x} + \boldsymbol{\mu} + \boldsymbol{\varepsilon} \quad (1)$$

where parameter $\mathbf{W}$ is a $D\times Q$ matrix that contains the factor loadings that achieves the projection, $\boldsymbol{\mu}$ allows the data model to have nonzero mean, and $\boldsymbol{\varepsilon}$ is an $\mathbf{x}$-independent noise process. By defining a prior distribution over $\mathbf{x}$, together with the distribution of $\boldsymbol{\varepsilon}$, equation (**1**) induces a corresponding distribution in the data space and the model parameters may be determined by maximum likelihood techniques.
The subspace defined by the columns of $\mathbf{W}$ will generally not correspond to the principal subspace of the data. However, probabilistic PCA (PPCA) is able to build a matrix $\mathbf{W}$ whose columns span the principal subspace of

the data for the case of isotropic noise $\varepsilon \sim N(0, \sigma^2 I)$,as we can see in [5].

PPCA proposes the following estimations for its parameters:

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{t}_n \tag{2}$$

where N is the cardinal of the data set, and:

$$\mathbf{W} = \mathbf{U}_q (\Lambda_q - \sigma^2 \mathbf{I})^{1/2} \mathbf{R} \tag{3}$$

where $\mathbf{U}_q$ is a matrix which comprises the q principal eigenvectors of the covariance matrix $\mathbf{S}$, with corresponding eigenvalues in the diagonal matrix Λ_q. $\mathbf{R}$ is an arbitrary orthogonal rotation matrix, and the noise variance, σ^2, is defined as:

$$\sigma^2 = \frac{1}{D-Q}\sum_{i=Q+1}^{D} \lambda_i \tag{4}$$

with $\lambda_{q+1}, \ldots, \lambda_D$ the smallest eigenvalues of $\mathbf{S}$.

The parameters $\mathbf{W}$ and σ^2 of the model can also be calculated iteratively by an expectation-maximization (EM) algorithm (as it is defined in [5]).

$$\mathbf{W}_{new} = \mathbf{SW}(\sigma^2 \mathbf{I} + \mathbf{M}^{-1}\mathbf{W}^T\mathbf{SW})^{-1} \tag{5}$$

$$\sigma^2_{new} = trace(\mathbf{S} - \mathbf{SWM}^{-1}\mathbf{W}_{new}^T)/D \tag{6}$$

where

$$\mathbf{M} = (\sigma^2 \mathbf{I} + \mathbf{W}^T\mathbf{W}) \tag{7}$$

It is required a lot of computational work to calculate the covariance matrix and extract the eigenvalues needed to define the parameters. It can be done in a direct way, which implies a complexity of $O(ND^2)$ and $O(D^3)$ respectively. However the cost can be reduced significantly if we use the EM algorithm when the dimension of the data is much more large than the number of latent variables, i.e., $D >> Q$, we evaluate trace($\mathbf{S}$) in $O(ND)$ and we apply the optimization

$$\mathbf{SW} = \frac{1}{n}\sum_n (\mathbf{t}_n - \boldsymbol{\mu})\{(\mathbf{t}_n - \boldsymbol{\mu})^T \mathbf{W}\} \tag{8}$$

with an associated complexity of $O(NDQ)$.

It has been proved that PPCA preserves the property of optimal linear least-squares reconstruction. In this sense, the best approximation to the original data vector is defined as

$$\mathbf{t}_{rec} = \mathbf{W}(\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T(\mathbf{t}_n - \boldsymbol{\mu}) + \boldsymbol{\mu} \tag{9}$$

3. PPCA Neural Model with a global-local competition.

We extend the classical competitive learning model and show a new neural network formed by a set of neurons with probabilistic PCA capacity, where the number of latent variables, i.e. basis vectors, needed to retain a particular amount of variance is inferred by the network itself. This type of network has a certain number of processing units or neurons. Each of one stores its own estimations of the parameters $\boldsymbol{\mu}$, $\mathbf{W}$ and σ^2 which will be utilized to describe the probabilistic PCA. The training is achieved in a batch mode, i.e., in each iteration, called epoch, all samples are presented to the network. Then a competitive process is carried out to select the *winning neuron* for every sample. After that, the neurons are updated according to the information provided by their new receptive field, i.e. the set of inputs for which the neuron is the winner. Finally, the retained variance of every cluster is analyzed. If the variance kept by the cluster is under a certain minimum, the dimensionality of the subspace must grow. On the other hand, if we conserve more variance than it is required we try to decrease the number of basis vectors. Forthcoming, we present in a detailed way the phases of the procedure mentioned above.

3.1 Competition among neurons

In a training epoch, all samples of the learning dataset are presented to the network. For each learning sample $\mathbf{t}_n$ a competition is held among the neurons and the unit c will be the winner for that sample if the value of the competition rule, i.e. the measure employed to assign the samples to the clusters, is the minimum:

$$c = \arg\min_j \left(r_j(\mathbf{t}_n) \right) \tag{10}$$

where $r_j(\mathbf{t}_n)$ used to be the Euclidean distance between the sample and the mean of cluster j in classical competitive neural networks(see [11]), or the reconstruction distortion achieved by the cluster j (see [3]).

We propose the competition rule

$$r_j(\mathbf{t}_n) = D_j(\mathbf{t}_n) + d_j(\mathbf{t}_n) \tag{11}$$

with $D_j()$ a function which expresses a measurement in the global input space, whereas $d_j()$ relies on some way to preserve the homogeneity in local subspaces.

The reconstruction distortion for sample $\mathbf{t}_n$ if we assign to cluster i has been selected as the global measurement:

$$D_j(\mathbf{t}_n) = \left\| \mathbf{t}_n - \mathbf{t}^j_{nrec} \right\|^2 \tag{12}$$

where $\mathbf{t}^j_{nrec}$ is calculated by the process unit following equation (9). This allows to represent the dataset very faithfully.

The neural model must be able to represent the input space with a high accuracy, but we must ensure the generalization property of the network is also high enough. Therefore, those samples which present small reconstruction distortion for neuron i but are too far from the mean of the cluster must be penalized in order to make

more difficult their belonging to the cluster i. Hence, our local measurement is defined as it follows:

$$d_i(\mathbf{t}_n) = \left\| \boldsymbol{x}_n^i \right\| \tag{13}$$

with $\boldsymbol{x}_n^i$ the vector $\mathbf{t}_n$ expressed in the reduced-dimensionality representation given by the latent variable model of neuron i, i.e. $\mathbf{t}_n^i = \mathbf{W}^{\mathrm{T}}(\mathbf{x}_n - \boldsymbol{\mu})$.

Once all competitions have finished, a new partition of the input dataset into M clusters is obtained where every neuron encompasses the samples which are best represented by it, according to the competition rule.

$$C_i = \left\{ \mathbf{t} \mid \forall j = 1,\ldots,M,\ r_i(\mathbf{t}) \le r_j(\mathbf{t}) \right\} \tag{14}$$

Then, the information of those new clusters is taken into account and the update of the units is carried out as it will be explained in the next subsection.

3.2 Neuron weights updating

PPCA requires the estimation of $\boldsymbol{\mu}$, $\mathbf{W}$ and σ^2 as we explained in section 2. The estimation of $\boldsymbol{\mu}$ is carried out like in the classical competitive neural network model where the synaptic weights of each neuron provide information about the centroid of the cluster the unit represents. After a training epoch the new cluster samples must be considered and the mean estimation must be adequately updated according to that information. Thus, let the new samples mean for the neuron i after training epoch n

$$\boldsymbol{\mu}_{new}(n) = \frac{1}{|C_i|}\sum_{\mathbf{t}\in C_i} \mathbf{t} \tag{15}$$

the mean estimation is adaptively modified with

$$\boldsymbol{\mu}_i(n+1) = (1-\eta(n+1))\boldsymbol{\mu}_i(n) + \eta(n+1)\boldsymbol{\mu}_{new}(n) \tag{16}$$

where $\eta(n+1)$ is the learning rate and shows the amount of knowledge that the neuron can extract from the new samples assigned to the cluster in opposition to its actual learned information. Initially, a new configuration of the cluster has a lot of influence on the mean estimation. However, the previous learning efforts are not forgotten and some of the preceding knowledge is maintained. As the learning process evolves the model focus mainly in its background information and new data is slightly used to refine the mean approximation.

If equation (16) is reformulated we get

$$\boldsymbol{\mu}_i(n+1) = \boldsymbol{\mu}_i(n) + \eta(n+1)(\boldsymbol{\mu}_{new}(n) - \boldsymbol{\mu}_i(n)) \tag{17}$$

where it is not difficult to see that geometrically the mean estimation is approaching to the real cluster mean.

On the other hand, we compute approximations to $\mathbf{W}$ and σ^2 for every neuron. These approximations are iteratively calculated by means of equations (5) and (6) of the optimized EM algorithm. The values needed to apply these equations (the trace of the covariance matrix and the matrix product $\mathbf{SW}$) can be easily obtained through an adaptive learning process, similar to the mean estimation. The trace of the covariance matrix $\mathbf{S}$ of the cluster i at training step n is

$$\mathrm{trS}_{new}(n) = \frac{1}{|C_i|}\sum_{\mathbf{t}\in C_i} trace\left((\mathbf{t}-\boldsymbol{\mu}_i)(\mathbf{t}-\boldsymbol{\mu}_i)^T\right) \tag{18}$$

Therefore, the approximation for the next epoch can be presented as follows:

$$\mathrm{trS}_i(n+1) = (1-\eta(n+1))\,\mathrm{trS}_i(n) + \eta(n+1)\,\mathrm{trS}_{new}(n) \tag{19}$$

In a similar way the estimation of the matrix product $\mathbf{SW}$ for neuron i can be derived from equation (8) as:

$$\mathbf{SW}_{new}(n) = \frac{1}{|C_i|}\sum_{\mathbf{t}\in C_i} (\mathbf{t}-\boldsymbol{\mu}_i)\left((\mathbf{t}-\boldsymbol{\mu}_i)^T\,\mathbf{W}_i(n)\right) \tag{20}$$

and we can use the same strategy for the estimation at the epoch $n+1$:

$$\mathbf{SW}_i(n+1) = (1-\eta(n+1))\mathbf{SW}_i(n) + \eta(n+1)\mathbf{SW}_{new}(n) \tag{21}$$

3.3 The explained variance method

The explained variance method considers a variable number of basis vectors $K_i(t)$, which is computed independently for each neuron i. This number reflects the intrinsic dimensionality of the data in the receptive field of the neuron, i.e. the set of inputs for which the neuron is the winner. Through the learning process the method ensures that a minimum amount of variance is conserved for each cluster in order to satisfy the level of accuracy chosen by the user. The number of principal components which are preserved must be modified consequently to reach that level at least.

In a local PCA method the worst representation error is obtained when all the components are ignored and the the mean is the only statistical information which is kept, i.e. when we lose all the variance relative to the directions of change of the samples. Equation (4) quantifies the lost variance per discarded dimension in a cluster subspace. Thus, the total cluster variance when no directions are kept, V_0, can be defined as:

$$V_0 = D\frac{1}{D}\sum_{p=1}^{D}\lambda_i^p = \sum_{p=1}^{D}\lambda_i^p \tag{22}$$

In any other situation the discarded variance, V_Z, depends on the number of basis vectors of the neuron:

$$V_Z = (D-Z)\frac{1}{D-Z}\sum_{p=Z+1}^{D}\lambda_i^p = \sum_{p=Z+1}^{D}\lambda_i^p \tag{23}$$

Our goal is to minimize the dimension of the subspace, which implies a more compressed representation of the data, but the model must maintain a minimum level of quality α. with respect to the maximum accuracy the method can achieve. Let $V_0 - V_Z$ be the amount of error (we must remember that the more variance is lost is more likely the samples are represented less accurately) eliminated when the cluster subspace preserves Z basis vectors then

$$K_i = \min\left\{Z \in \{0,1,...,D\} \mid V_0 - V_Z \geq \alpha \ V_0\right\} \quad (24)$$

Substitution of (22) and (23) into (24) yields

$$K_i = \min\left\{Z \in \{0,1,...,D\} \mid \sum_{p=1}^{D} \lambda_i^p - \sum_{p=Z+1}^{D} \lambda_i^p \geq \alpha \sum_{p=1}^{D} \lambda_i^p\right\} \quad (25)$$

From PCA we know that the sum of variances equals the trace of the covariance matrix **S**, then equation (25) can be simplified as follows:

$$K_i = \min\left\{Z \in \{0,1,...,D\} \mid \sum_{p=1}^{Z} \lambda_i^p \geq \alpha \ trace(\mathbf{S}_i)\right\} \quad (26)$$

If we add the time instant n we get the equation to be used in practice:

$$K_i(n) = \min\left\{Z \in \{0,1,...,D\} \mid \sum_{p=1}^{Z} \lambda_i^p(n) \geq \alpha \ \mathrm{tr}\mathbf{S}_i(n)\right\} \quad (27)$$

4. Experimental results

We have designed a set of experiments to test the representation ability of the neural network, referred as CPPCA in the rest of the section. We have compared its results with the ones provided by the Mixture of Probabilistic Principal Components Analysis (MPPCA) presented by Tipping and Bishop (see [**5**]). The measure of performance we have selected is the *mean normalized squared projection error* (MNSPE)

$$MNSPE = \frac{1}{N}\sum_{n=1}^{N} \frac{\|\tilde{\mathbf{t}}_n\|^2}{\|\mathbf{t}_n\|^2} \quad (28)$$

with $\tilde{\mathbf{t}}_n = \mathbf{t}_n - \mathbf{t}_{nrec}$ the projection error of the considered model for that sample.

The dataset comes from the VizieR service [**12**], which is an information system for astronomical data. In particular, we have selected the Table 6 of the Complete near-infrared and optical photometric CDFS Catalog from Las Campanas Infrared Survey [**13**]. We have extracted 22 numerical features from 10,000 stars. Hence, we have 10,000 sample vectors. These data have been normalized in order to cope with the variability and the very heterogeneous scaling of the original data.

We have carried out experiments with Q values from 1 to 18 , in the case of MPPCA, and with explained variance 0.1, 0.3, 0.5, 0.7,0.8, 0.9 and 0.95 in other case. All these values of Q and explained variance have been combined with 4, 8, 16, 32 and 64 neurons. A linear learning rate with an initial value $\eta(0) = 0.9$ has been used for CPPCA. It has been trained along 30 epochs. MPPCA model only uses 10 epochs because more training time shows no significant performance improvements.

The plots of the error *MNSPE* versus the number of basis vectors Q are shown in Figures 1 to 5. We have calculated the mean over the dimensionality of the clusters in order to present the neural network results in the figures.

In all the tests, the curve that links all the results of our neural network is slightly nearer the coordinate axis than the curve of MPPCA model, which means that our model gets better representation of the data.

If we fixed an amount of explained variance, we can notice that as the number of neurons we employ are growing up the dimensionality of the subspaces is normally diminishing. The cause is that the presented model is able to make more homogeneous clusters which allow to conserve less vector basis to maintain at least the required user precision α..

In general, the more processing units the network contains, the less the reconstruction error is obtained. The level of quality specified by the user through the parameter α is also very important to determine the final reconstruction error. As the figures show, when the percentage of preserved variance is high, especially if it is near one, the distortion is very small because all statistical information is kept and the precision mainly depends on the PCA method implemented by the network. However if the amount of information which we retain is small, i.e. we preserve few vector basis or none, this discarded knowledge contribute to increase the final error too much.

5. Conclusion

A new unsupervised neural model has been presented. It combines competitive learning with probabilistic principal components analysis and shows the ability to select the exact dimension each cluster needs according to a quality criterion specified by the user. Therefore, there is no need for a prior knowledge about the dataset which we apply the model on, and consequently, the adaptive capacity of the network is not restricted.

Experimental results which have been presented measure the performance of our approach when we use it to represent a multidimensional dataset. Our model slightly outperforms the Mixture of Probabilistic Principal Components Analysis, which is a well-known model usually taken as a reference and whose authors are Tipping and Bishop. CPPCA offer a more condensed representation of the dataset without loss in quality, i.e. it is able to generate the same distortion error but using lower dimensionality in the subspaces of the clusters.

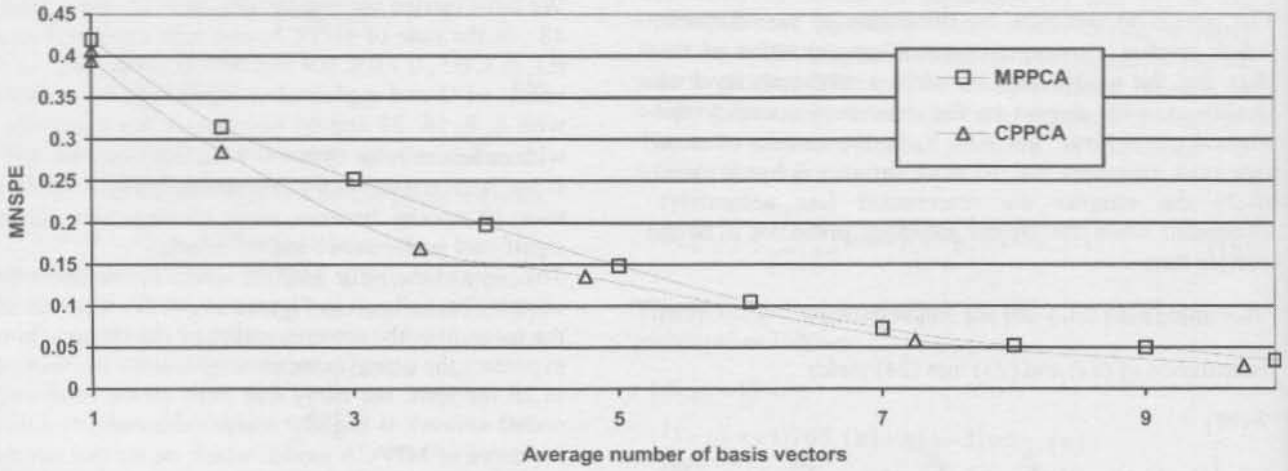

Fig. 1. Network performance with four neurons.

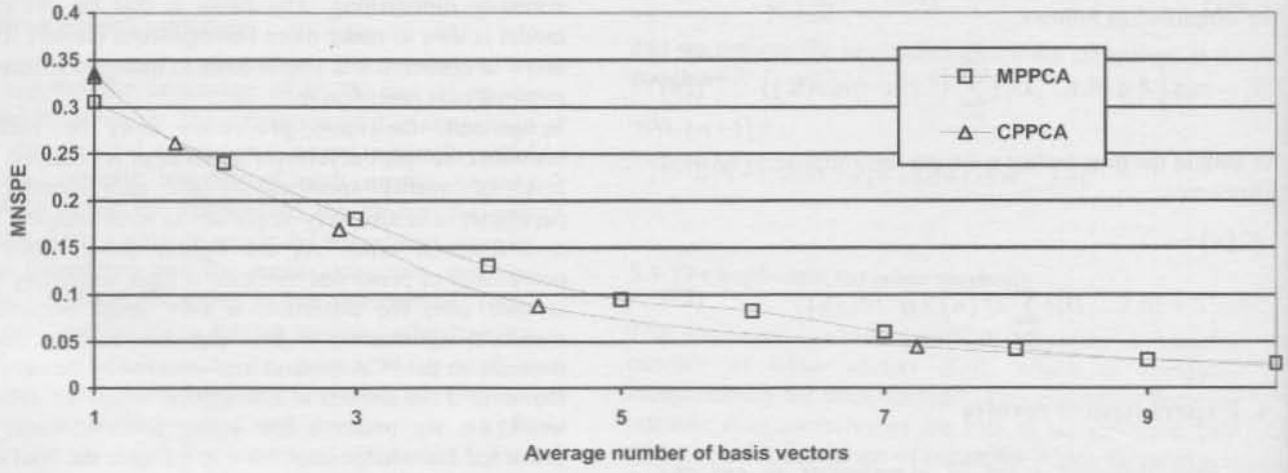

Fig. 2. Network performance with eight neurons.

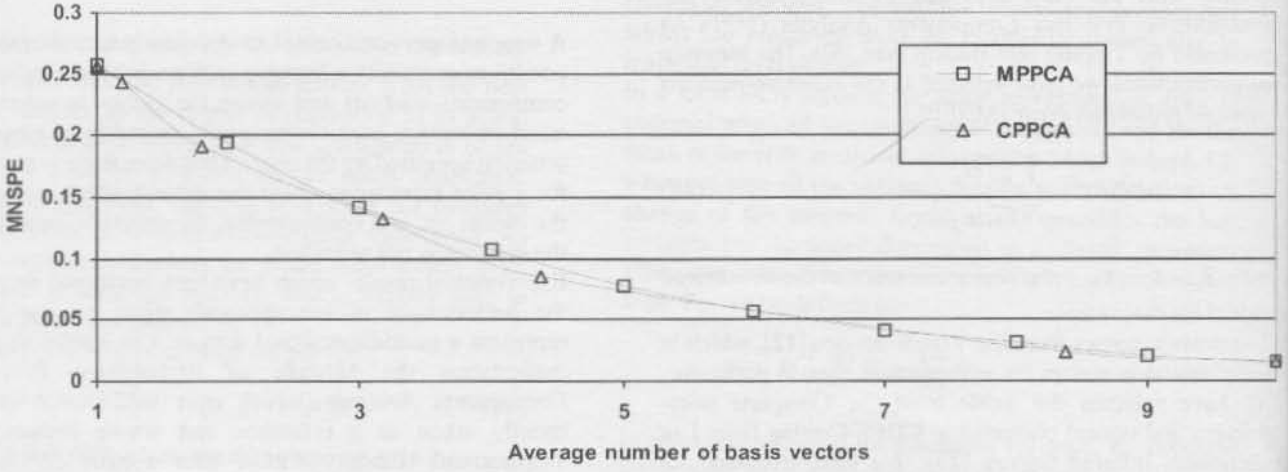

Fig. 3. Network performance with sixteen neurons.

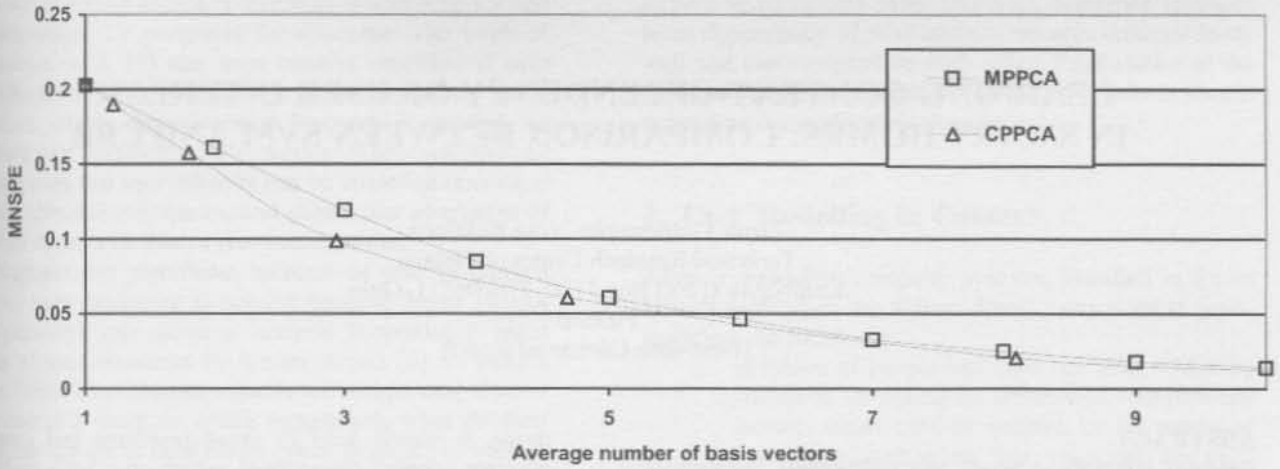

Fig. 4. Network performance with thirty two neurons.

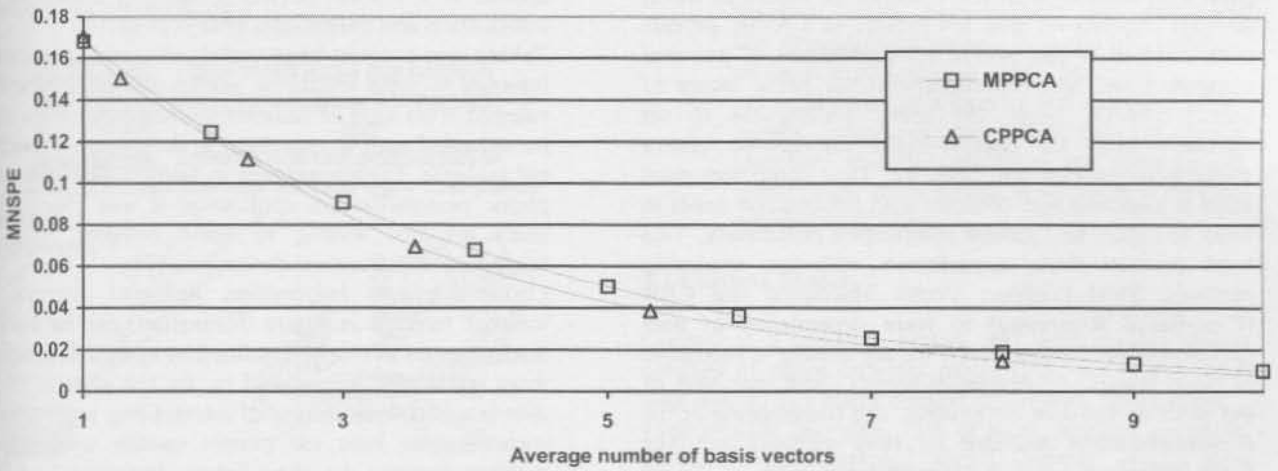

Fig. 5. Network performance with sixty four neurons.

References

[1] Jolliffe, I.T.: Principal Component Analysis, Springer-Verlag, Berlin, (1986).

[2] Kendall, M.: Multivariate Analysis, Charles Griffin&Co, London (1975).

[3] Kambathla, N. Leen, T.K.: Dimension Reduction by Local Principal Component Analysis, Neural Computation, **9** (7), (1997) 1493–1516.

[4] Roweis, S., Ghahramani, Z.: A Unifying Review of Linear Gaussian Models, Neural Computation, **11**, (1999) 305–345.

[5] Tipping, M.E., Bishop, C.M.: Mixtures of Probabilistic Principal Components Analyzers, Neural Computation, **11**, (1999) 443–482.

[6] Saghri, J.A., Tescher, A.G., Reagan, J.T.: Practical Transform Coding of Multispectral Imaginery, IEEE Signal Processing Magazine, (January 1995) 32–43,.

[7] Tretter, D., Bouman, C.A.: Optimal Transforms for Multispectral and Multilayer Image Coding, IEEE Trans. Image Processing, **4** (3), (1995) 296–308.

[8] Bishop, C. M., Svenson, M., & C.K.I. William, GTM: The generative topographic mapping, Neural Computation, 10(1), (1998), 215-234.

[9] Besse, P., PCA stability and choice of dimensionality, Statistics & Probability Letters, *13*(5), (1992), 405-410.

[10] Minka, T. P., Automatic choice of dimensionality for PCA, Advances in Neural Information Processing Systems, *13*, (2001), 598-604.

[11] Ahalt, S.C., Krishnamurthy, A.K., Chen, P., Melton, D.E.: Competitive Learning Algorithms for Vector Quantization. Neural Networks, 3 (1990) 277-290.

[12]VizieR service [online]. Available at: http://vizier.cfa.harvard.edu/viz-bin/VizieR (March 29, 2004)

[13] Chen, H.-W., et al., Early-type galaxy progenitors beyond z=1, Astrophysical Journal, *560*, 2001, L131.