X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), 8 a 11 de Novembro de 2011, Fortaleza, Ceará

A Parallel PCA Neural Network Approach for Feature Extraction

Patrícia R. Oliveira, Luciano A. Digiampietri, Willian Y. Honda, Marcone C. Pereira
School of Arts, Science and Humanities - University of São Paulo, São Paulo – SP – Brazil
{proliveira, digiampietri, kio, marcone}@usp.br

Abstract – Principal Component Analysis (PCA) is a well known statistical method that has successfully been applied for reducing data dimensionality. Focusing on a neural network which approximates the results obtained by classical PCA, the main contribution of this work consists in introducing a parallel modeling for such network. A comparative study shows that the proposal presents promising results when a multi-core computer is available.

Keywords – principal component analysis, feature extraction, paralell neural network.

1 INTRODUCTION

In pattern recognition problems, dimension reduction is the process of decreasing the number of variables under consideration, and it can be divided into feature selection and feature extraction methods. Feature selection approaches try to find an optimal subset of the original variables by eliminating features with little or no predictive information. On the other hand, feature extraction techniques map the original multidimensional space into a space of reduced dimensionality. This means that the original feature space is transformed by applying, for instance, a linear transformation.

One of the most commonly used linear technique for dimensionality reduction is the Principal Component Analysis (PCA), which transforms the data in such a way that the variance in the lower-dimensional representation is maximized. In the classical statistical approach for performing PCA, the data correlation matrix is constructed and its eigenvectors are computed. The eigenvectors corresponding to the largest eigenvalues (the principal components) can be used as the basis of the transformed subspace [1,2] . Alternatively, neural network models can also find the principal components for a problem by simply computing much less complicated operations, such sums and multiplications throughout an iterative process.

Dimension reduction techniques are intensely required in image processing applications since the typical high dimensionality of this kind of data restricts the choice of image processing methods. Besides reducing the number of features to be processed, PCA can lead to the additional benefit of removing noise from the data, as such noise are usually concentrated in the excluded dimensions [3]. Although PCA is the best linear method (in mean-square sense) to project the data into a lower-dimensional subspace, it can be quite expensive to compute, depending on the image resolution and number of images to be processed [4].

The aim of this paper is to present an efficient parallel PCA neural network approach, to run in multi-core computers, for feature extraction. The proposed approach can be seen as an extension of the adaptive neural model developed by Rubner and Tavan [5].

The main motivation of our work is to take advantage of the popularization of multi-core personal computers and workstations. The proposed model was evaluated with experimental studies and its performance was compared, in terms of quality of the lower-dimensional images and computational cost, with the sequential PCA neural network and the traditional PCA, executed by the Minitab Statistical Software ^{TM}.

The remain of this paper is organized as follows. Section 2 discusses some recent approaches also developed in attempting to reduce PCA computational cost. The fundamentals of classical statistical PCA are briefly presented in Section 3. Section 4 gives the architecture and algorithm for sequential and parallel PCA neural network. In Section 5, the experimental results of this work are discussed. Finally, Section 6 presents the conclusions and future work.

2 RELATED WORK

Bingham and Mannila [4] compared different methods for image data dimensionality reduction using the criteria of amount of distortion caused by each method and its computational cost. They showed that the Random Projection (RP) technique does not distort the data significantly more than PCA and, for low compression rates, RP and PCA give better results than those produced by Discrete Cosine Transform (DCT). They also measured the number of float point operations needed when using RP, PCA and DCT in dimensionality reduction, concluding that PCA is significantly more burdensome than RP or DCT.

Aiming to overcome the drawbacks in traditional PCA, Ye et al. [3] have proposed a dimensionality reduction scheme, called Generalized PCA (GPCA), which works directly with images in their native state, as two-dimensional matrices. Such scheme has been tested in experiments on databases of face images, that showed that GPCA is superior to PCA in terms of quality of the compressed images, query precision, and computational cost.

Several parallel implementations of traditional PCA have been introduced by Yang et al. [6], who have investigated such proposals considering time speed and resulting compression performance. They have showed that parallel implementations using an eigenspace merging approach have lower speed performance than other based on covariance matrix merging.

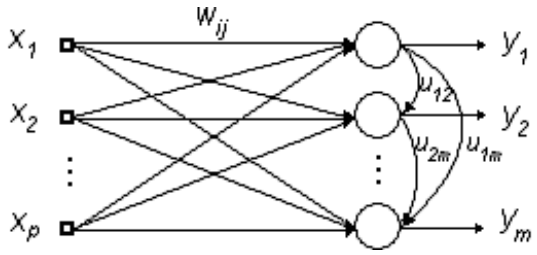

Figure 1: Sequential PCA Neural Network Architecture

3 PRINCIPAL COMPONENT ANALYSIS (PCA)

Essentially, the classical PCA aims to solve an eigenvalue problem:

$$C_x a_j = \lambda_j a_j, \quad for\ j = 1, 2, ..., p \tag{1}$$

where C_x is the original data covariance matrix, λ_j is an eigenvalue of C_x and a_j is the eigenvector corresponding to the eigenvalue λ_j. Next, the calculated eigenvalues can be increasingly ordered:

$$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p. \tag{2}$$

The principal components can, then, be computed according to the equation:

$$Z_j = a_j^T X = X^T a_j, \quad for\ j = 1, 2, ..., p \tag{3}$$

where Z_j is the $j\text{-}th$ principal component and X represents the original data set.

An important property of PCA is that the variances of principal components are the eigenvalues of matrix C. Therefore, dimensionality reduction can be obtained by performing PCA and by keeping only the components with highest variance.

4 PCA NEURAL NETWORKS

The **Sequential Adaptive PCA Neural Network** [5] uses an unsupervised learning process that is based on variations of the Hebbian learning rule. Its architecture, shown in Figure 1, consists of p input and m output units, organized in such way that the output unit i is connected to the output unit j with connection strength u_{ij}, if and only if $i < j$. For this model, it has been proved that the synaptic weight vector w_j converges to the $j\text{-}th$ eigenvector of the data covariance matrix, considering an ordering scheme by decreasing eigenvalues.

The output $y_j(n)$ of neuron j at time n produced in response to the set of inputs x_i, for $i = 1, 2, ..., p$, is given by [7]:

$$y_j(n) = \sum_{i=1}^{p} w_{ij}(n) x_i(n) + \sum_{k=1}^{j} u_{kj} y_k(n). \tag{4}$$

The synaptic weights w_{ij} between the input and output layers are updated in accordance to the Hebbian learning rule, that is,

$$\begin{gathered} \Delta w_{ij}(n+1) = \eta x_i y_i + \beta \Delta w_{ij}(n), \\ for\ i = 1, 2, ..., p\ and\ j = 1, 2, ..., m, \end{gathered} \tag{5}$$

where η is a learning constant and β is the *momentum* term. The lateral synaptic weights are adjusted according to the anti-Hebbian learning rule:

$$\begin{gathered} \Delta u_{kj}(n+1) = \mu y_k yj + \beta \Delta u_{kj}(n), \\ for\ k < j\ and\ j = 1, 2, ..., m, \end{gathered} \tag{6}$$

where μ is another positive learning parameter.

It has been proved that the internal weights of a neuron j in the sequential PCA neural network will only converge after all the internal weights associated to the $j-1$ neurons have already reached convergence [8]. Therefore, it is possible to develop a **Parallel PCA Neural Network** focusing on each neuron in an individual manner, while preserving the interdependence among them.

For constructing these individualized schemes, one can decompose the sequential PCA neural network architecture into smaller structures, each of them oriented by one output neuron. This idea is illustrated in Figure 2 for the first neuron, which finds the basis for the first principal component of the data. This neuron has to be considered in a special way, since this is the only one that does not receive any influence from the others in the output layer, but that contributes for all the other output results.

The second individualized structure is oriented by the neuron related to finding the second principal component, as illustrated in Figure 3. It can be seen that this neuron is partially dependent from the previous one and, thus, it is expected that it converges after the previous structure has been converged.

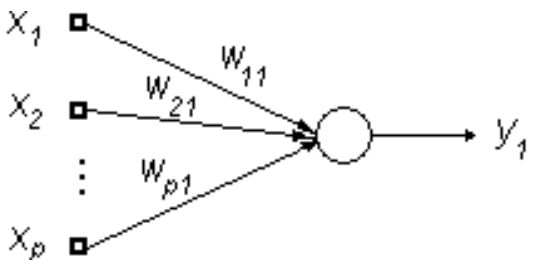

Figure 2: Individualized scheme for the first output neuron

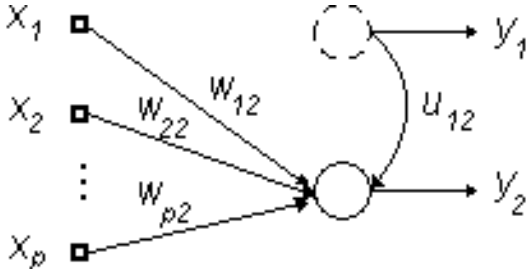

Figure 3: Individualized scheme for the second output neuron

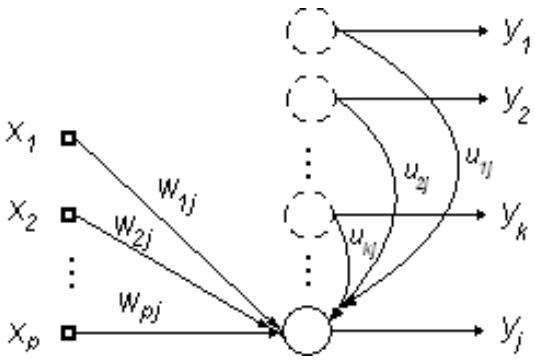

Figure 4: Individualized scheme for the j-th output neuron

Generalizing the idea, the individualized structure for the j-th neuron can be stated as seen in Figure 4.

For modeling the PCA neural network in a parallel fashion, one could now simply consider each individualized sub-network as running in a different thread. In this case, each thread would provide the computation for the basis of one principal component. However, since there is a partial interdependence among the neurons running in different threads, it turns out to be needed the use of synchronization mechanisms in order to guarantee the following restrictions: (i) that a neuron j does not finish its processing before all the $j-1$ neurons has been done; (ii) that a neuron j does not begin running before all the $j-1$ neurons has been started. The synchronization among the threads were done using semaphores.

5 EXPERIMENTAL RESULTS

The tests were realized in an Intel Core 2 computer with 2.0 GB of RAM and Ubuntu 9.10 operation system. The input data was composed of 20 images randomly chosen from the Microsoft Research Cambridge Object Recognition Image Database [9]. The original size of the images was 640x480 pixels, but, for this case study, they were reduced to 160x120 pixels to simplify the computation.

The preprocessing task involved the arbitrary choice of the number of eigenvectors and the division of the image in patches corresponding to the selected value. This number represents the number of neurons of the neural network. For the tests presented here, the number of eigenvectors was set to 10 and each image was divided in 1920 blocks of 10 pixels, producing a matrix with 1920 rows and 10 columns. This matrix was normalized and each cell in the matrix had its value decreased by the average of the matrix values.

The experiments run over two different implementations: the sequential and the parallel. For each image, both implementations were executed using as parameters the following number of cycles: from 500 to 5,000 ranging from 500 and from 10,000 to 50,000 ranging from 5,000 cycles. For each execution we stored the eigenvectors, the principal components, the execution time, and the Mean Square Error (MSE) calculated comparing the original image with the image reconstructed using the principal components. The following analyses will consider the average results of the 20 images used in this case study.

Figures 5 and 6 presents the MSE versus the number of cycles used in the training of the neural network. Each image was reconstructed using only 7 principal components and the error was calculated comparing the reconstructed image and the original one. It is important to observe that the parallel implementation converges quicker than the sequential. The number of cycles in the training range from 500 to 5,000 and from 5,000 to 50,000 cycles.

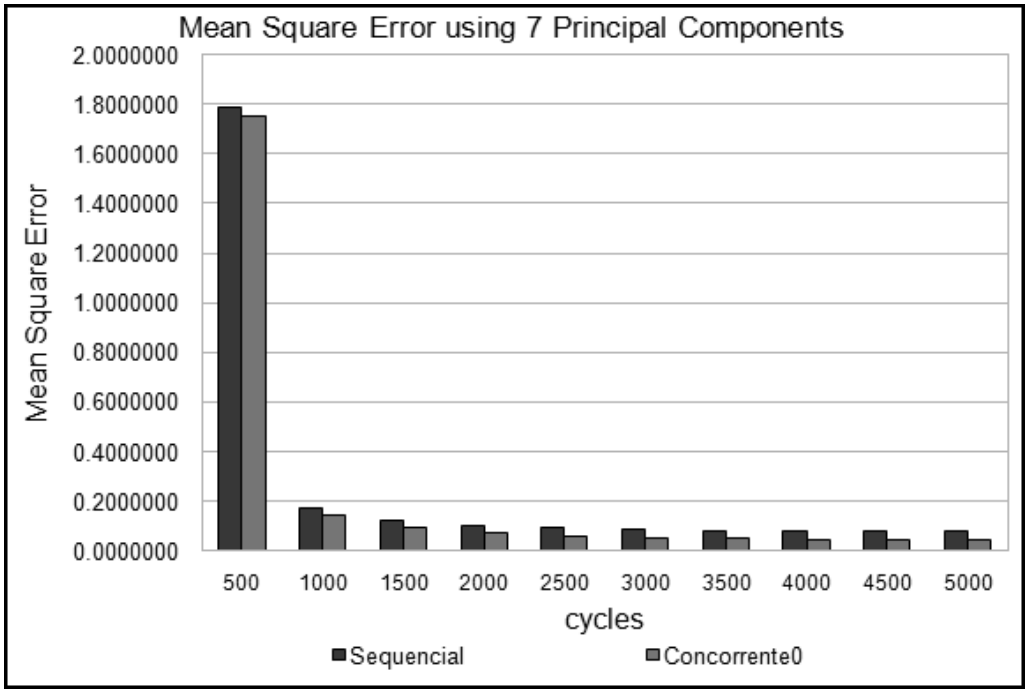

Figure 5: Mean Square Error using 7 principal components

As described in the literature [7, 8], the neural network for calculating PCA will converge when trained with enough number of cycles. To illustrate that, we compared the MSE of the implemented neural networks (sequential and our parallel proposal) with the results produced by Minitab^{TM} Statistical Software. Figure 7 presents this comparison. The neural networks were trained with 50,000 cycles. As expected, the three techniques converged for the same results.

The execution time in milliseconds of the sequential and parallel approaches is showed in Table 1. The last column of the table present the relation between the time spent in the parallel approach and the time spent in the sequential one.

Figure 8 presents the reconstruction of one of the 20 images used in this case study. The reconstruction used from 1 to 10 principal components produced by the parallel algorithm trained with 50,000 cycles. The legend of each image contains the

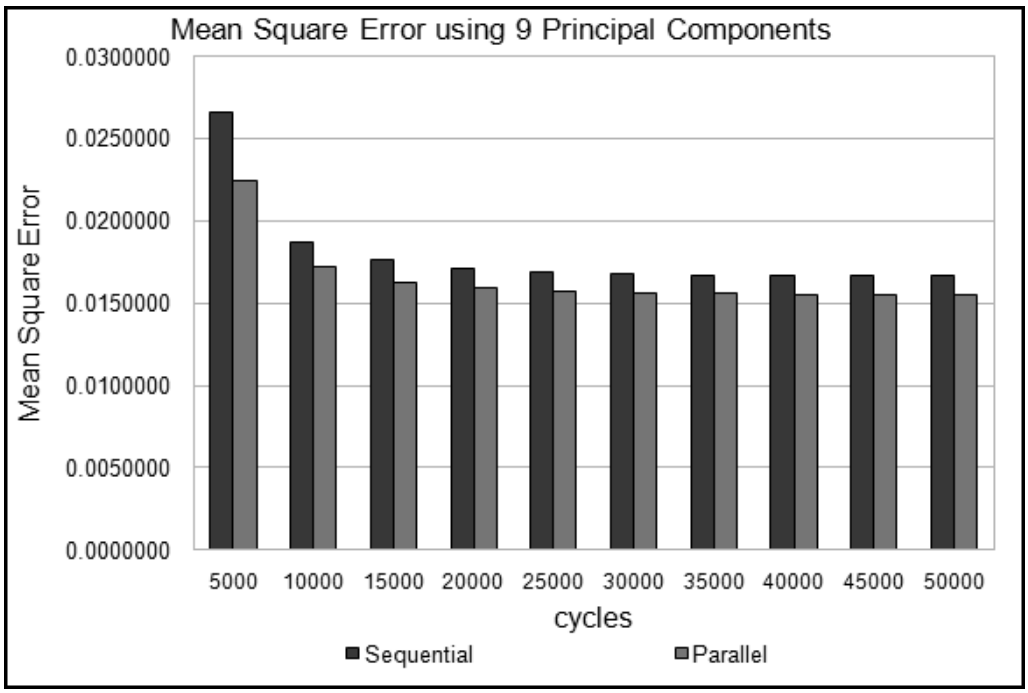

Figure 6: Mean Square Error using 7 principal components

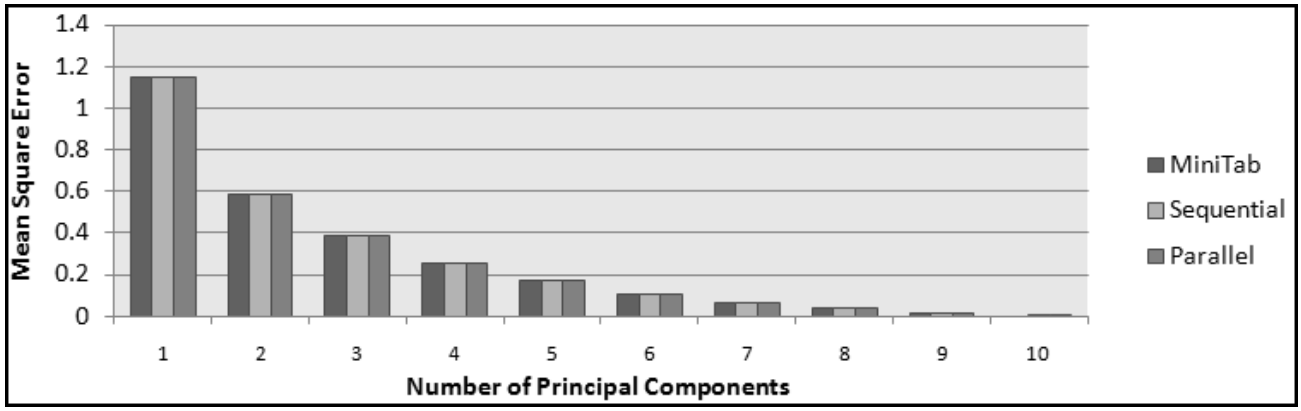

Figure 7: Mean Square Error: comparison with Minitab^{TM}

number of components used in the reconstruction and the Mean Square Error of the reconstruction when compared with the original image.

The tests presented in this section were realized in a dual core computer, and show that the parallel algorithm run in less then 73% of the execution time of the sequential one, and, for the same amount of training cycles, the parallel approach presented a smaller MSE than the sequential one. It occurs because in the parallel approach running using x cores, the neuron j will start to execute only when neuron $j - x$ finishs to execute. So, it will start its learning process using converged data from the neurons 1 to $j - x$. In contrast, in the sequential approach, all neurons are learning at the same time using data from the previous neurons which hasn't converged yet.

6. CONCLUSION AND FUTURE WORK

This article presented the proposal of a parallel PCA neural network, which is based on an extension of the adaptive neural model developed by Rubner and Tavan [5]. A comparative study discussed the results obtained by such approach in comparison to those obtained by the sequential PCA neural network and the Minitab statistical software.

The experimental results showed that the parallel network run faster than its sequential version and, for the same number of training cycles, it presented smaller MSE values than those provided by the traditional one.

As future work, experiments should be carried out in computers with different numbers of cores in order to verify the scalability of the proposed approach.

# of cycles	Sequential	Parallel	Parallel/Sequential
10000	124216	89618	72.1%
20000	247179	178700	72.3%
30000	370528	267741	72.3%
40000	494159	358797	72.6%
50000	619018	448456	72.4%

Table 1: Execution Time in Milliseconds

Figure 8: Reconstruction of one image using from 1 to 10 Principal Components

REFERENCES

[1] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice-Hall, Inc., Englewood Cliffs, NJ, fourth edition, 1998.

[2] A. Gorban, B. Kegl, D. Wunsch and A. Zinovyev. *Principal Manifolds for Data Visualisation and Dimension Reduction*. Springer, 2008.

[3] J. Ye, R. Janardan and Q. Li. "GPCA: an efficient dimension reduction scheme for image compression and retrieval." In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 354–363, 2004.

[4] E. Bingham and H. Mannila. "Random projection in dimensionality reduction: Applications to image and text data". In *in Knowledge Discovery and Data Mining*, pp. 245–250. ACM Press, 2001.

[5] J. Rubner and P. Tavan. "A Self-Organizing Network for Principal Component Analysis". *Europhysics Letters*, vol. 10, no. 7, pp. 693–698, 1989.

[6] H. Yang, Q. Du, W. Zhu, I. Banicescu and J. E. Fowler. "Parallel Data Compression for Hyperspectral Imagery". In *Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International*, volume 2, 2008.

[7] J. Mao and A. K. Jain. "Artificial neural networks for feature extraction and multivariate data projection". *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 296–317, March 1995.

[8] S. Haykin. *Redes Neurais: Princípios e prática*. Bookman, second edition, 2001.

[9] Microsoft. "Microsoft Research Cambridge Object Recognition Image Database". http://research.microsoft.com/research/downloads/Details/ b94de342-60dc-45d0-830b-9f6eff91b301/Details.aspx (ass of 2010-01-22), 2005.