A Cellular Neural Network as a Principal Component Analyzer

Chao-Hui HAUNG, Wee-Kheng LEOW, and Daniel RACOCEANU

***Abstract*—In this paper, A configuration of Cellular Neural Network (CNN) is introduced to implement Principal Component Analysis (PCA). CNN is a parallel computing paradigm. Many researchers considered it as the next generation universal machine and developed so-called CNN universal chips. Based on the capability of CNN, an alternative PCA implementation named Principal Component Analyzing Cellular Neural Network (PCACNN) is proposed. PCA is used to reduce the dimensions of a given dataset in order to extract the principal information of the given dataset. In decades, many researchers presented their investigations based on PCA in order to improve the performance and/or to attack some open issues in specific fields. In this paper, PCA is implemented based on the architecture and capabilities of CNN. Consequently, the computing performance of PCA can be improved as long as the CNN architecture can be realized.**

I. Introduction

CELLULAR NEURAL NETWORK, also known as CNN, has been considered as an alternative parallel computing architecture. Since Chua and Yang first introduced CNN, it has been widely applied on many areas[1], [2]. CNN has been considered as a simple but powerful parallel computing paradigm. It is an locally connected neural network and has evolved into a paradigm for a type of array. The updating rule of a cell in CNN is only involving it's neighboring cells. Thus, the computing performance can be excellent and meanwhile, the neural structure remains simple.

Some investigations introduced their CNN realization. For example, the ACE4K chip [3], [4], [5]. Other researchers even considered CNN as a next generation universal machine [6], [7], [8]. In those CNN machines, the information on the cell array can be update more than 4000 times per second. In the last version of the chips, it can even update the array more than 16000 times per second. Based on their works, some interesting researches were initiated for the sake of implementing specific applications. Usually, the performance of those applications can be improved because of the capabilities of CNN and the machines. However, since the architecture of CNN is very different from the traditional algorithmic architecture, how to implement them on CNN is a challenge.

In this research, an alternative implementation of Principal Component Analysis (PCA) is investigated. Based on the architecture and capabilities of CNN, the performance of PCA can be improved. PCA is an algorithm that can compute the eigenvectors of the covariance matrix of the given dataset corresponding to its largest eigenvalues, and project the samples of the multivariate process on the eigenvectors in order to obtain a principal component set. It was introduced by Karl Pearson in the beginning of the 20th century. Nowadays, it remains a famous tool for data analysis, simplification, and reduction in many fields.

PCA is defined as an orthogonal linear transformation which can transform a given input to a new coordinate system in order to maximize the variance by the projection of the input. Thus, the greatest variance by any projecting of the data comes to lie on the first component, the second greatest variance on the second component, and so on. PCA has been considered as the optimum transform for a given dataset in least square terms.

PCA is frequently used to reduce the dimensions of a dataset by retaining the variance of a dataset. The dataset contributes to most of its variance, by keeping lower-order principal components and ignoring higher-order ones, or vice versa. This is depends on the rule of PCA in the design of a system.

However, the performance of mathematical implementation of PCA decreases as the scale of a given input increases. To overcome this problem, several neural network structures and their learning rules have been proposed, there are the so-called PCA Neural Networks (PCANNs). The pioneering work of Prof. Oja and his research team introduced online estimation of principal components by linear neural networks. Consequently, many interesting implications on unsupervised learning theories and applications to neural signal processing are inspired [9], [10], [11], [12], [13].

Followed Oja, Sanger *et al.* extended PCANN by the well-known Sanger's rule, also known as Generalized Hebbian Algorithm (GHA). GHA has become an important tool in many fields since it is simple and effective.

Adaptive Principal Component Extractor (APEX) was presented by Kung *et al.* for the sake of conquering the problems in the recursive computation of the principal components of a vector stochastic process and performance improvement [14], [15].

Followed their works, many other researchers improved the performance of PCANN. However, some of their investigations have a non-ignorable trade-off: the quality of PCANN becomes worse as the computational complexity is reduced. Fiori indicated the computing complexity of different types of PCANN, including GHA, APEX, y^2-APEX, and 0-APEX [16]. This fact implies that the computing consumption is highly related to the numbers of the given input.

Thus, in this paper, an alternative solution is considered, named Principal Component Analyzing Cellular Neural Net-

Chao-Hui HUANG, Wee-Kheng LEOW, and Daniel RACOCEANU are with the Department of Computer Science, National University of Singapore; and Image Perception, Access & Language (IPAL) French-Singaporean Joint Laboratory (email: {huangch,leowwk,danielr}@comp.nus.edu.sg).

This research is supported by A*STAR SERC 052 101 0103 (NUS R-252-000-319-305).

978-1-4244-3553-1/09/$25.00 ©2009 IEEE

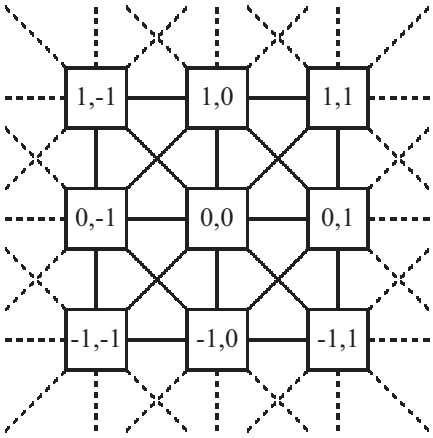


Fig. 1: An example of Cellular Neural Grids. Each cell is only connected to its neighboring cells.

work (PCACNN). One special feature of Cellular Neural Network is that the scale of the array in Cellular Neural Network is not related to the computing time consumption since the Cellular Neural Network is processing in a parallel paradigm. Based on this feature, the performance of PCA can be improved. This investigation is also indicating a pathway to implement PCA on analogical circuits (see Fig.(3)) [1].

In this paper, first, the fundamental concepts of PCA and CNN are introduced. Next, the architecture of PCACNN is presented. Some interesting characteristic of PCACNN are also investigated. Third, some comparisons with the existing technologies are drawn. Finally, the conclusion follows.

II. Methodology

A. Cellular Neural Network (CNN)

CNN is in a grid-like structure (see Fig.(1)). A cell in CNN only communicates with its neighboring cells. The output value of the cell is affected by the neighboring cells. The output values of its neighboring cells are also affected by the cell (see Fig.(2) and Fig.(3)). This interactive updating rule results the the power of CNN [1].

The equations of a CNN cell are defined as the follows [1]:

- State equation:

$$\begin{aligned} C\frac{d}{dt}v_{x_{ij}} &= -\frac{1}{R_x}v_{x_{ij}}(t) + \\ &\sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l)v_{y_{kl}}(t) + \\ &\sum_{C(k,l)\in N_r(i,j)} B(i,j;k,l)v_{u_{kl}}(t) + I, \\ &1 \le i \le M, \text{ and } 1 \le j \le N. \end{aligned} \quad (1)$$

- Output equation:

$$\begin{aligned} v_{y_{ij}}(t) &= \frac{1}{2}(|v_{x_{ij}}(t)+1| - |v_{x_{ij}}(t)-1|), \\ &1 \le i \le M, \text{ and } 1 \le j \le N. \end{aligned} \quad (2)$$

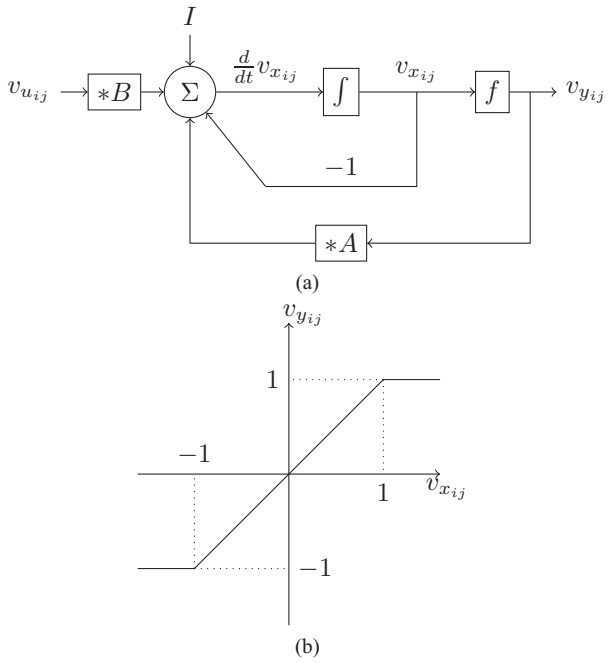


Fig. 2: The architecture of CNN is shown in (a) and along with the dynamic route of state in (b).

- Input equation:

$$v_{u_{ij}} = E_{ij}, 1 \le i \le M, \text{ and } 1 \le j \le N. \quad (3)$$

- Constraint equations:

$$\begin{aligned} &|v_{x_{ij}}(0)| \le 1, |v_{u_{ij}}| \le 1, \\ &1 \le i \le M, \text{ and } 1 \le j \le N. \end{aligned} \quad (4)$$

- Parameter assumptions:

$$A(i,j;k,l) = A(k,l;i,j), \quad (5)$$
$$1 \le i \le M, \text{ and } 1 \le j \le N.$$
$$C > 0 \text{ and } R_x > 0. \quad (6)$$

The state equation (1) and the output equation (2) are also illustrated in Fig.(2).

B. PCA Neural Network (PCANN)

For the sake of implementing the PCA based on CNN, first the existing approaches of PCA Neural Networks (PCANN) are investigated, including Oja's Rule and Sanger's Rule.

1) Oja's Rule: Oja's rule is a linear feedforward neural network model for unsupervised learning with applications primarily in PCA. It defines the update rule in presynaptic weights w_i which given the output y of a neuron to its inputs x_i. That is

$$\Delta w_i = \eta y(x_i - y w_i). \quad (7)$$

The output y is then updated as

$$y = \sum_{i=1}^{N} w_i x_i, \quad (8)$$

where N is the length of the given input.

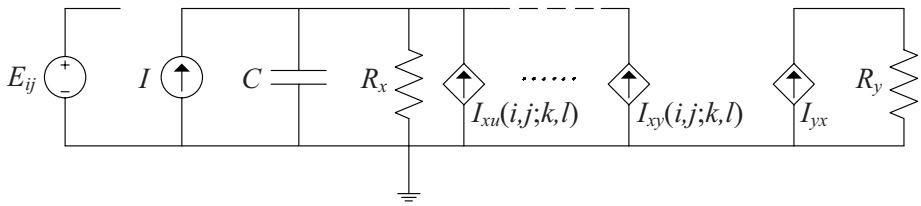


Fig. 3: An example of a cell circuit. C is a linear capacitor, R_x and R_y are linear resistors; I is an independent voltage source; $I_{xu}(i,j;k,l)$ and $I_{xy}(i,j;k,l)$ are linear voltage-controlled current sources with the characteristics $I_{xy}(i,j;k,l) = A(i,j;k,l)v_{y_{k,l}}$ and $I_{xu}(i,j;k,l) = B(i,j;k,l)v_{u_{k,l}}$ for all $C(i,j) \in N_r(i,j)$; $I_{yx} = (1/R_y)f(v_{x_{i,j}})$ is a precewise-linear voltage-controlled current source with its characteristic $f(\cdot)$ as shown in Fig.(2b); E_{ij} is an independent voltage source [1].

2) Sanger's Rule: Sanger's Rule, also known as Generalized Hebbian Algorithm (GHA), is similar to Oja's rule in its formulation, except it can be applied to networks with multiple outputs.

$$\Delta w_{ij} = \eta(y_j x_i - y_j \sum_{k=1}^{j} w_{ik} y_k), \tag{9}$$

The output y_j is then updated as

$$\begin{aligned} & y_j = \sum_{i=1}^{N} w_{ij} x_i, \text{ for each } j, \\ & 1 \le j \le M, \end{aligned} \tag{10}$$

where N is the length of the given input and M is the number of principal components.

C. Derivation of Principal Component Analyzing Cellular Neural Network (PCACNN)

Assume CNN is operated in linear region, (1) can be reformatted as

$$\begin{aligned} C\frac{d}{dt}v_{x_{ij}} &= -\frac{1}{R_x}v_{x_{ij}}(t) + \\ &\sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l)v_{x_{kl}}(t) + \\ &\sum_{C(k,l)\in N_r(i,j)} B(i,j;k,l)v_{u_{kl}}(t) + I, \\ &1 \le i \le M, \text{ and } 1 \le j \le N. \end{aligned} \tag{11}$$

The templates can be declared as

$$\begin{aligned} A(i,j;k,l) &\equiv a(k-i,l-j) \text{ and} \\ B(i,j;k,l) &\equiv b(k-i,l-j). \end{aligned} \tag{12}$$

They are defined as

$$a(k-i,l-j) = \begin{cases} -y^2(j) + \frac{1}{\eta}, & k = i \text{ and } l = j \\ -y(j)y(l), & k = i \text{ and } l < j \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

and

$$b(k-i,l-j) = \begin{cases} y(j), & k = i \text{ and } l = j \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

where

$$1 \le i \le M, \text{ and } 1 \le j \le N. \tag{15}$$

Next, let $v_{x_{ij}}(t) \equiv w_t(i,j)$, $dv_{x_{ij}}/dt \equiv \Delta w_t(i,j)$, $u_{ij} \equiv x_t(i)$ for each j, $C = 1/\eta$, $R_x = \eta$, and $I = 0$, the state equation (1) can be reorganized as

$$\begin{aligned} \frac{1}{\eta}\Delta w_t(i,j) &= -\frac{1}{\eta}w_t(i,j) + \\ &\sum_{l=1}^{N} a(k-i,j-l)w_t(i,l) + \\ &\sum_{l=1}^{N} b(k-i,j-l)x(i) \\ &= -\frac{1}{\eta}w_t(i,j) + \\ &\sum_{l=1}^{j} (-y(j)y(l)w_t(i,l)) + \\ &\frac{1}{\eta}w_t(i,j) + y(j)x(i), \\ &1 \le i \le M, \text{ and } 1 \le j \le N. \end{aligned} \tag{16}$$

Finally, the state equation becomes

$$\begin{aligned} \Delta w_t(i,j) &= -\sum_{l=1}^{j} \eta y(j)y(l)w_t(i,l) + \eta y(j)x(i) \\ &= \eta(y(j)x(i) - y(j)\sum_{l=1}^{j} y(l)w_t(i,l)), \\ &1 \le i \le M, \text{ and } 1 \le j \le N. \end{aligned} \tag{17}$$

Note that (17) is exactly the update rule of the General Hibbean Algorithm (see (9)). In (17), $x(i)$ is the *i*-th input, $y(j)$ is the *j*-th output, the M is the number of the inputs and the N is the number of outputs. $w_t(i,j)$ is the weight for i-th input and j-th output at time t. $w_t(\cdot,\cdot)$ approaches to principal components as computing time goes to infinity.

The $y(\cdot)$ is updated as

$$\begin{aligned} & y(j) = \sum_{i=1}^{N} w_t(i,j)x(i), \text{ for each } j, \\ & 1 \le j \le M, \end{aligned} \tag{18}$$

such that the templates are also updated.

For example, assume a design has five inputs and it requires the first two principal components, the definition

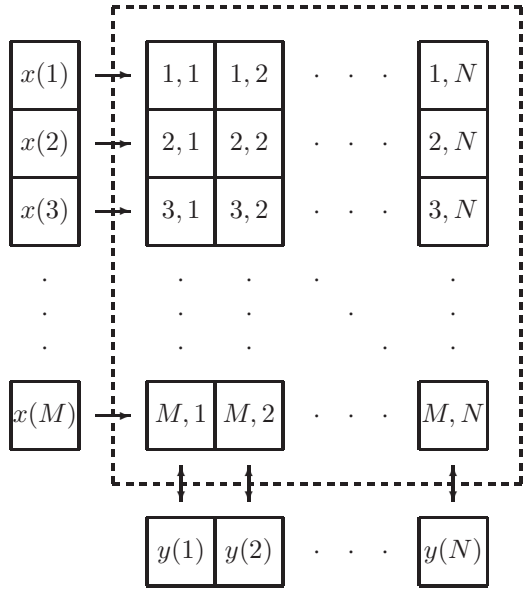


Fig. 4: The cell array of the proposed architecture. The $x(\cdot)$'s are the given input, the $y(\cdot)$'s are the eigenvalues, the area that is surrounded by a dot-line is the cell array $w(\cdot,\cdot)$. Each column approaches the principal component as the computing time goes to infinity.

of the cell array would be

$$w_t(i,j), 1 \leq i \leq M, \text{ and } 1 \leq j \leq N. \tag{19}$$

where $M = 5$ and $N = 2$. The templates would be

$$A = \left[\begin{array}{ccc} -y(j)y(j-1), & -y^2(j) + \frac{1}{\eta}, & 0 \end{array} \right], \tag{20}$$

and

$$B = y(j), \tag{21}$$

where $1 \leq j \leq N$. The size of the templates will be increased if more principal components are required.

The proposed architecture is represented in Fig.(4). In the picture, the $x(\cdot)$'s are the given input, the $y(\cdot)$'s are the eigenvalues, the area that is surrounded by a dot-line is the cell array $w(\cdot,\cdot)$. Each column approaches the principal component as the computing time goes to infinity.

D. Locally Connected PCACNN

In the proposed architecture, the length of the template A can be calculated by $2N - 1$, where N is the desired number of the principal components. In order to reduce the complexity of the neural structure, a shorter length of template A can also be a option. Since for each cell in the array, all of the connected cells are its neighboring cells. Thus, it is named Locally Connected PCACNN (LC-PCACNN). The reduction of the connections implies the fact that the system might be divergence. The examples shown in the experiments.

III. Experiments

Fig.(5) presents the comparisons of the simulation results between APEX and PCANN with different numbers of principal components. In the figures, both of the approaches present that the errors are convergent to minima. However, as the number of the principal components are increased, PCACNN results faster convergence. Meanwhile, In the paradigm of CNN, the array size is not related to the computing complexity since it is parallel computing. In another words, whatever the array size is, ideally, the time consumption of the computing would be the same.

Fig.(6) shows the results of Locally-Connected PCACNN. Note that in this experiment, the desired number of principal components of the APEX model are 7. In this experiment, the number of the connected cells are from 1 to 11, respectively.

The proposed architecture was implemented in a microscopic image cell detection system. The system is built for the breast cancer diagnosis. In clinical breast cancer grading, the pathologists need to evaluate the number of mitotic cells in order to make medical decisions. This system can help the pathologists to count the mitotic cells under the microscope. One result is presented in Fig.(7), where the light boxes on the screen indicate the locations of the mitotic cells. As the results, the number of the mitotic cells can be evaluated.

IV. Conclusion

In this paper, an alternative implementation for PCA based on parallel computing paradigm is proposed. The PCACNN can parallelly compute and update the cell array and consequently, the PCA model can be implemented by analogical circuits. Meanwhile, the quality of the computation remains. In this investigation, the proposed model is compared with APEX, which currently is widely used since it has better performance. The experiments indicates the fact that the performance of PCACNN is better than the existing technology if the number of the principal components are bigger. Manwhile, based on the parallel compuging paradigm of CNN, PCACNN can also present excellient performance with a realized CNN architecture.

On the other hand, in the presented model, the templates of the CNN is so-called non-linear templates. The implementation of those kinds of templates are rather difficult. Thus, how to simplify the templates can be one of the future works.

References

[1] L. O. Chua and L. Yang, "Cellular neural networks: theory," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1257–1272, 1988.

[2] ——, "Cellular neural networks: applications," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1273–1290, 1988.

[3] Analogic-Computers, *ALADDIN CNN Software Library (Templates and Algorithms)*. Budapest: Analogic Computers Co.Ltd., 2000.

[4] ——, *ALADDIN CNN Software Library for ACE4K Chip (Templates and Algorithms)*. Budapest: Analogic Computers Co.Ltd., 2000.

[5] G. Liñán, S. Espejo, R. Domínguez-Castro, and A. Rodríguez-Vázquez, "Ace4k: an analog i/o 64 x 64 visual microprocessor chip with 7-bit analog accuracy," *International Journal of Circuit Theory and Applications (CTA)*, vol. 30, pp. 89–116, 2002.

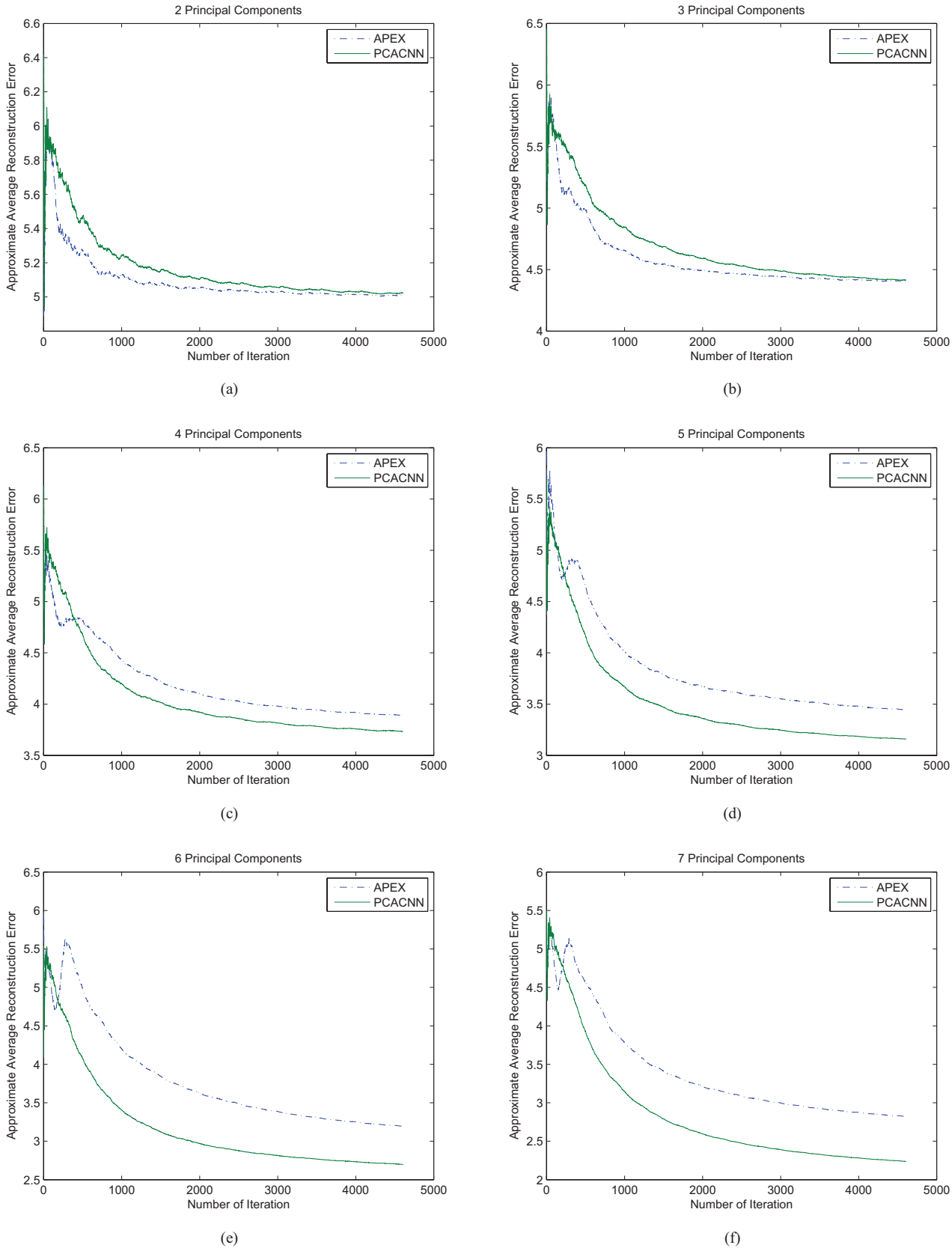
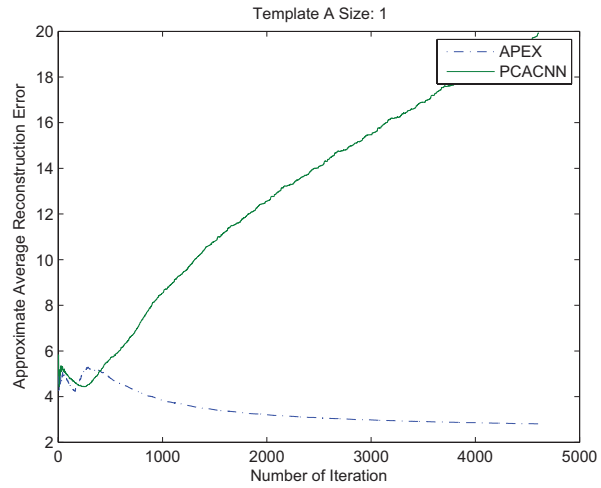
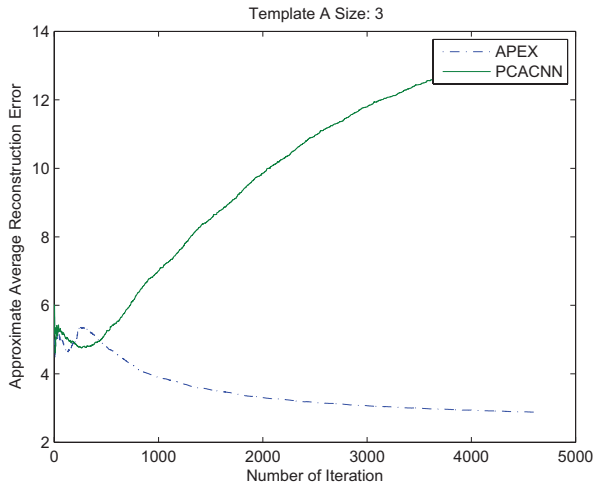


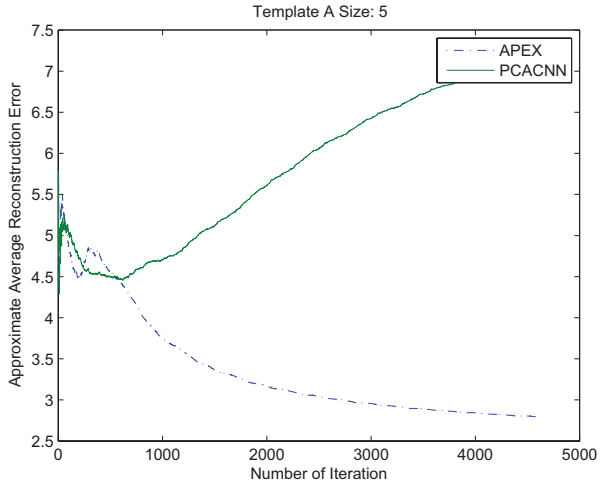
Fig. 5: The figures illustrate comparisons between APEX and PCACNN with different number of principal components.



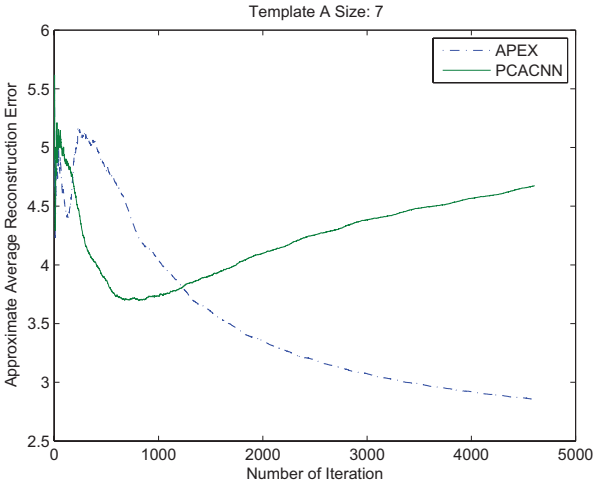
(a)



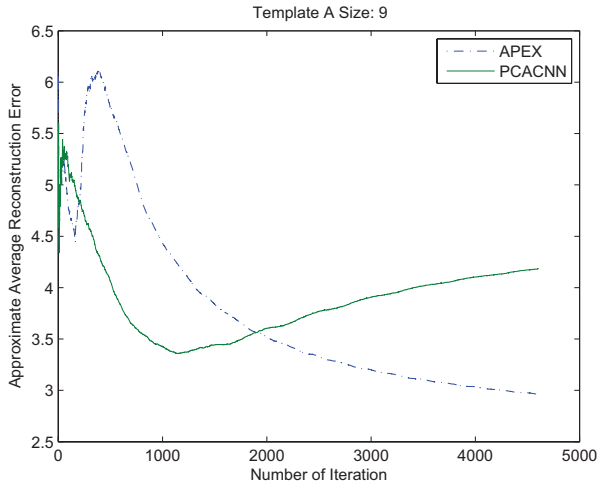
(b)



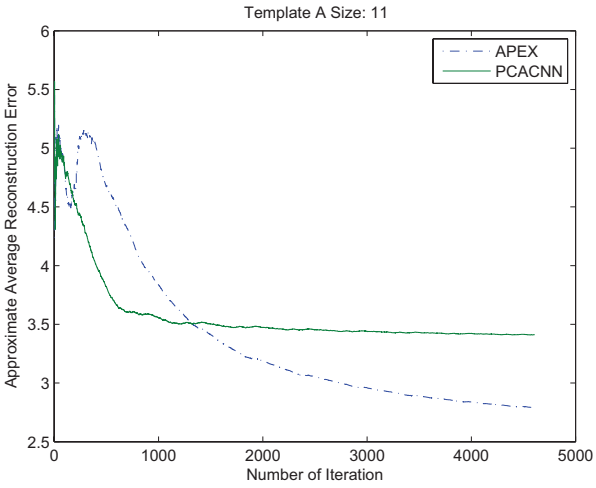
(c)



(d)



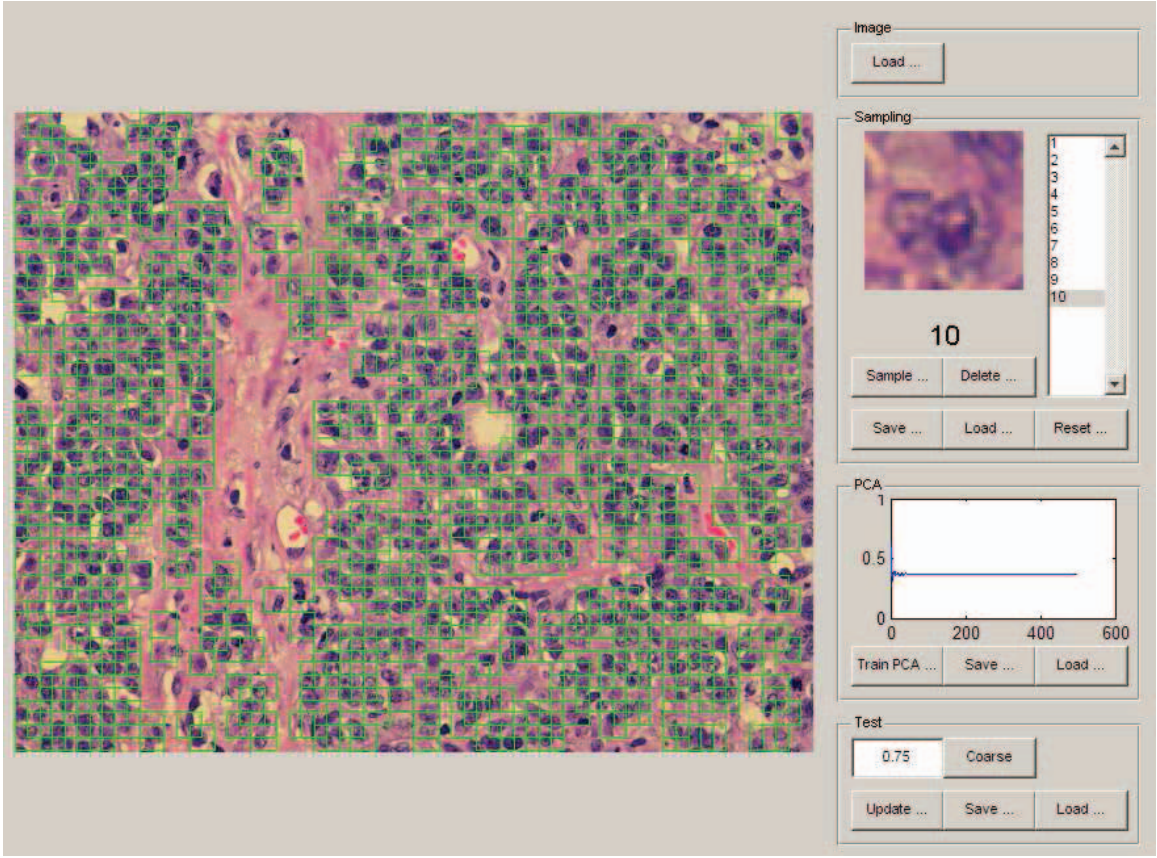
(e)



(f)

Fig. 6: The results of the proposed method for locally connected templates.

[6] T. Roska and L. O. Chua, "The cnn universal machine: An analogic array computer," *IEEE Trans. Circuits Syst. - II*, vol. 40, pp. 163–173, 1993.
[7] C. Rekeczky, I. Szatmári, and P. Földesy, "Computing on silicon with trigger-waves: experiments on cnn-um chips," in *IEEE International Symposium on Circuits and Systems 2001, ISCAS 2001*, Sydney, 2001.
[8] H. Kim, T. Roska, H. Som, and I. Petras, "Analog addition/subtraction on the cnn-um chip with short-time superimposition of input signals," *IEEE Trans. Circuits Syst. - I*, vol. 50, no. 3, pp. 429–432, 2003.
[9] P. Baldi and K. Hornik, "Learning in neural networks: a survey," *IEEE Trans. on Neural Networks*, vol. 6, no. 4, pp. 837–858, 1995.
[10] K. Hornik and C.-M. Kuan, "Convergence analysis of local feature extraction algorithms," *Neural Networks*, vol. 5, pp. 229–240, 1992.
[11] J. Karhunen, "Optimization criteria and nonlinear pca neural networks," in *International Conference on Neural Networks (ICNN)*, 1994, pp. 1241–1246.
[12] F. Palmieri and J. Zhu, "Anti-hebbian learning in topologically constrained linear networks: a tutorial," *IEEE Transactions on Neural Networks*, vol. 4, no. 5, pp. 748–761, 1993.
[13] A. Weingessel and K. Hornik, "Svd algorithms: Apex-like versus subspace methods," *Neural Processing Letters*, vol. 5, pp. 177–184, 1997.
[14] S. Kung and K. Diamantaras, "A neural network learning algorithm for adaptive principalcomponent extraction (apex)," in *1990 International Conference on Acoustics, Speech, and Signal Processing, 1990. ICASSP-90.*, vol. 2, 1990, pp. 861–864.
[15] K. Diamantaras and S. Kung, *Principal component neural networks: theory and applications*. J. Wiley and Sons, 1996.
[16] S. Fiori and F. Piazza, "A comparison of three pca neural networks," in *European Symposium on Artificial Neural Networks, SEANN'1999*. Bruges, Belgium: D-Facto public, 1999, pp. 275–280.



(a)

Fig. 7: An application of the proposed method is presented. The PCACNN is trained to detect the mitotic cells. The light boxes indicate the locations of the mitotic cells.