Journal of Theoretical and Applied Information Technology
15th November 2016. Vol.93. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

P-CC-NN: PARALLEL CASCADE CORRELATION NEURAL NETWORK METHODS FOR PATTERN RECOGNITION APPLICATIONS USING MULTICORE TECHNIQUES

[1]KHALID MOHAMMAD JABER, [2]SOKYNA M. ALQATAWNEH
[1,2]Al-Zaytoonah University of Jordan, Faculty of Science and Information Technology, Amman, Jordan.
E-mail: [1]drkjaber@yahoo.com/k.jaber@zuj.edu.jo , [2]S.qatawneh@zuj.edu.jo

ABSTRACT

This paper presents a multi-core programming model that implements the cascade correlation neural networks technique, to enhance the classification phase of any pattern recognition system. It is based on combining the strengths of both approaches in order to construct an efficient Parallel Cascade Correlation Neural Network (P-CC-NN) system. In this work a complex case of pattern recognition system which is a 3D facial data has been used to examine the proposed system and ensure its effectiveness, experimental results are presented using 360 3D facial images, each image contains 96 distinguishable features. Results show significant improvement in execution time about 31 minutes (4.6 times speedup) in comparison with 146.5 minutes for serial time, this topology generated an accuracy of 94 %. This work is the first approach to handle the classification challenges for different pattern recognition applications using multi-core techniques.

Keywords: *Parallel, Cascade Correlation Neural Network, Pattern Recognition, Facial Image.*

1. INTRODUCTION

Computer technology has recently made improvements that ask for having better security techniques which have raised an interest in biometrics application, that and also in artificial neural networks, this is in order to be able to solve all kinds of classification problems. The reason a biometric cannot be forgotten is because it is a physical property; the special thing about it is that it has the potential to identify a person that is in very different settings. Which is in opposite contrast to, neural networks are a very powerful tool which can reproduce extremely complicated non-linear dependencies [1].

The facial recognition as a biometric are varied and vast and is an applications that for example, it is possible to make it useful as a security measure such as being used to keep control of documents in a way of security measure for example with passports and digital chip and pin cards. Furthermore, it could be used as a security ID for computer systems. Additionally it could even be used for computer games.

There have been great changes and in this field of automatic face recognition major advances have been made but there is still the problem that it is still difficult and regarded as a problem to solve and to continue attracting the research that is strong enough from the different disciplines and the areas that include the machine learning, pattern recognition and computer graphics [2].

The Training process of neural networks, especially for very large training datasets must be done as soon as possible because of the fact that a neural network for complicated and multi-dimensional problems usually means using very large amounts of training examples with the use of hundred, thousand or even millions of patterns. This means the training can take up to weeks and even months to reach the desired accuracy. It is also in this way that finding an optimal neural network configuration can require a certain amount of cross-validation experiments, which can be also very time consuming.

In today's market the best way of obtaining significant increases in CPU performance is by using a multi-threaded and multicore CPU with shared memory. It is expected that a large growth in performance is expected from them in the near future from more hardware threads and cores per CPU [3]. By using the new CPUs researches have started to concentrate all their attention on

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

parallelizing a variety of computational intelligence algorithms [4-7].

The main objective of this research is to provide an efficient classification system by constructing a robust parallel cascade correlation neural networks system using a multicore programming technique, this system will overcome the problems of cost, speed and execution time for different pattern recognition applications. This work is the first approach to handle the classification challenges for different pattern recognition applications using multi-core techniques.

The rest of this paper is organized as follows: In Section 2 we present a brief literature on parallel cascade correlation neural networks and its related work with facial recognition systems. The methodologies of this work are introduced in section 3. Section 4 discusses the experimental results of this research. Finally, the conclusion and future directions for our work are presented in Section 5.

2. LITERATURE REVIEW

Automatic Pattern Recognition is cast as a template matching problem, where classification has to be performed in a high-dimensional space. Since the higher the dimensionality of the space, the more the computation is needed to find a match. The combination between multi-core programming and neural networks technique can be seen as a good solution to such problem. This section will focus on assessing the state-of-the-art of both methods.

2.1 Cascade Correlation Neural Network

There are many benefits and advantages of the Neural Network (NN), it has proven to be a very useful tool which can also be used for solving many real-life problems. However, the issue is that it is because of its efficient implementation of the NN, it usually requires long training sessions [8] which can vary depending on the training vector and on the topology of the NN [8]. While it is known that the NN has been widely studied and evaluated in the pattern recognition application, Cascade-Correlation Neural Networks (CCNN) learning approach has received literally little attention and focus in pattern classification problems, specifically in 3D recognition applications. This come after it was found that the CCNN topology achieves and gives good performance in terms of the convergence time and optimum topology for other pattern recognition applications [2]. It is now in this network that the first layer has connecting weights to the input layer and each subsequent layer has weights which are connecting it to all previous layers including the input layer.

Threw the experiments in which the number of input nodes as well as the hidden nodes and the related topologies were changed so that we could find the best ones, we found that in the CCNN experiments, this was done in order to find the best inputs and their related topologies. First, it grows the network on demand, this means that it only adds new neurons and this is only when they can help in solving the problem. Second, the new neurons are added and trained individually this way each one can get the proper attention and also it makes it easier to eliminate the problems that have occurred.

Fahlman at el. [8] claim that cascade correlation algorithm is attractive for parallel implementation because the candidate units do not need to interact which means it can be trained independently. Few previous efforts have been made to enhance cascading correlation neural networks and few others have reported parallel cascading correlation neural network in different fields. For the first time, P-CC-NNs was used to recognize and classify a 3D facial dataset. In order to evaluate the performances of the learning system, the Jack-knife technique was employed with the use of 80% randomly selected samples for training and the remaining 20% for testing. After conducting extensive experiments using different number of images, it was found that P-CC-NNs provided more accurate results for face recognition than using other machine learning techniques [2].

2.2 Parallel Cascade Correlation Neural Network

In the past few decades we have had the equipment and ability to make use of automatic systems for face recognition. One of the reasons for growing interest in this topic is the huge potential and possibility in the applications for face recognition systems. One way could be to give training in regard to the neural networks but that is too time consuming. Therefore, a real effort is being made and dedicated to the training time in to the different areas and fields in regard to the hardware architectures. Such as, Altaf et al. [9] who proposed two techniques of parallelizing back propagation neural networks using multicore programming which is based on multithreading and general-purpose computation on graphics

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

processing units (GPGPU). And also Xavier et al. [10] who came up with the idea of the parallel training of a back-propagation neural network using CUDA on GPU. Both sets was then given for implementation and were tested with two standard benchmark data sets which were provided by PROBEN1. The two parallelization strategies on a cluster computer; training example and node parallelism using MPI approach was presented by Pethick et al. [11].

A proposal was made by David German [12] in which he presented that a project proposal titled computing hardware for accelerated training of cascade-correlation neural networks by parallelization of multiply-accumulate operations implemented on an field–programmable gate array (FPGA) in communication with a host PC. However, the problem is that the author did not make very clear some of the points and his ideas, for example, he did not mention their benchmark to evaluate their method such as speedup, overhead, etc., and he also didn't mention the parallel technique in details.

It was proposed by Lyle et al. [13] that the scalable massively parallel artificial neural networks for pattern recognition application. In this approach, the MPI used to parallelize the C++ code. However, each layer is distributed equally over all processors, which mean they used the data decomposition approach to parallelize the algorithm. There are further versions of parallel neural network which uses the task decomposition paradigm for cluster system where they duplicates the full neural network at each cluster node was presented by Dahl et al. [14]. Their system was implemented using MPI library. Schuessler et al. [15] proposed a method for parallelization of neural network training based on the backpropagation algorithm and implemented it using two different multithreading techniques (OpenMP and POSIX threads) applicable to the current and next generation of multithreaded and multi-core CPUs.

Moreover, Ingrid et al. [16] proposed the parallel training data of recurrent cascade correlation learning architecture (RCC) to recognize Japanese phonemes using a method called time-slicing. However, it was intended by the authors that in parallel RCC there will be a large number of training patterns or the training set should be divided into smaller chunks which are to be trained separately and sequentially, this is done in such a pattern that it goes from the simplest to the most complicated one. Therefore, the authors did not use the standard parallel computing concepts and models such as shard memory, distributed memory. Furthermore, they did not use parallel programming approaches such as multithreading, MPI or GPGPU.

3. METHODOLOGY

The aim of this research is to construct a powerful parallel computing system using a multicore programming model that implements 3D facial data, the proposed system will overcome the problems of cost, speed and time consuming for different pattern recognition applications. The effect of adapting the cascade correlation neural network to enhance the classification phase of recognition system is increased length of time it takes, in this section two multicore technique approaches that use shared memory model are presented as follow:

3.1 The Parallel Cascade Correlation Neural Network for 3D Facial Recognition (P-CC-NN)

In this section we will propose a Parallel Cascade Correlation Neural Network (P-CC-NN) to improve the classification phase of any recognition

Function Name	Calls	**Total Time**	Self Time*	Total Time Plot (dark band = self time)
network.train	100	88.790 s	0.133 s	
trainlm	1100	86.764 s	0.052 s	
network.train>trainPerWorker	100	86.690 s	0.015 s	
trainlm>train_network	100	86.661 s	27.598 s	
nnCalcLib>nnCalcLib.perfsJEJJ	4464	22.785 s	2.780 s	
perfsJEJJ	4464	20.005 s	0.251 s	

Figure 1. A snapshot of the MATLAB profiling function for CC-NN algorithm (small data set)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

system. In order to calculate we have used tl MATLAB profiling function, this allowed us to s how much time was spent on each function of tl correlation neural network program, which th allowed us to make a calculation. Furthermore, see things such as the highest time functions in C NN algorithms which were produced by the use this function you can look at Fig.1 which shows snapshot of these readings. It is threw this we c identify which of the functions of CC-NN ha consumed most of the time. In addition to this t the use of the MATLAB profiling function we c keep records of the information of things such number of calls, parent functions, execution tim total time, child functions and self-timer which shown in Figure 1. From this we can distingui that the majority of the time is taken up by CC-N is the network train which represents the tra cascade correlation in neural network. Due to th information, the P-CC-NN system was proposed.

So with that the first approach was implemented using a MATLAB implicit parallel for-loop (parfor), and it is with the number of threads that we identified and reserved with the parpool command. Nonetheless, the MATLAB parallel for-loop (parfor) dynamically divided the iteration of the loop; this was done based on the amount of threads which were made available by the parallel pool. Therefore, from this we see that if the amounts of threads are equal to the amount of loop iterations, then each worker must perform one iteration of the loop. However, if there are greater iterations than threads, it will be required that some workers will have to perform more than one loop iteration; if so this means that a worker may receive more iterations at one time to reduce the time of communication.

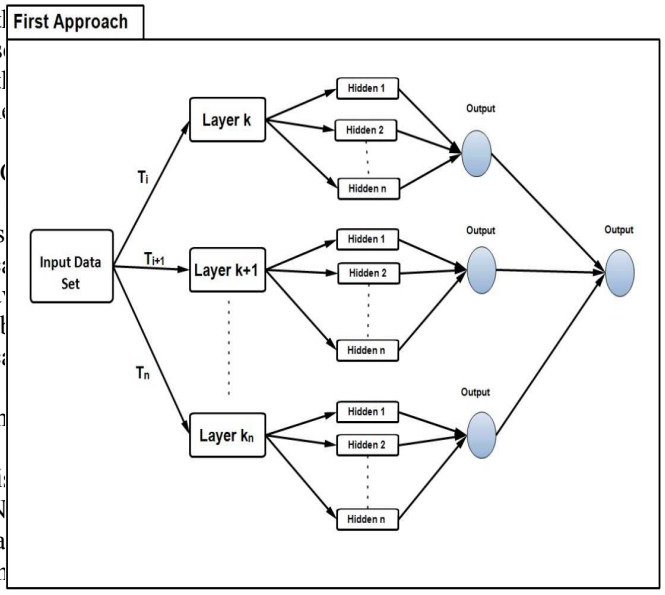

Figure 2. Parallel Mechanism proposed for P-CC-NN using implicit parallelism model

For a presentation of the implementing of P-CC-NN which is using explicit parallel model that is referred to as create independent jobs to represent the second approach as illustrated in Figure 3. In this approach, the P-CC-NN has gained more control in comparison to the first approach, where the difference was that every step had to be done and controlled manually in order to create and run the job. The following steps are required in order to apply this approach to the P-CC-NN:

1. A practical cluster must be identified. Therefore, in this step the P-CC-NN decides after inspection of which cluster it shall use. The majority of most clusters have a set maximum number of workers they can apply and use.
2. When the first step of the process is complete, the P-CC-NN produces and defines the thread tasks that are divided amongst all the concealed nodes and layers that are represented by $CCNN_{training}$= *{(Hi, layer Xi),(Hi+1, layer Xi).....(Hn, layer Xn)}* in order to be evaluated by the thread workers whilst the running of the job. At first, the tasks for each of the hidden nodes are created in order to make the cascade correlation layers. The tasks are represented by T= *{Ti, Ti+1,...Tn}*. Each thread can processes more than hidden nodes depending on number of threads; e.g. a thread number 1 (T_1) processes the several hidden nodes from layer 1 and layer 2 and (T_2) processes the several nodes from layer 2 and so on.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

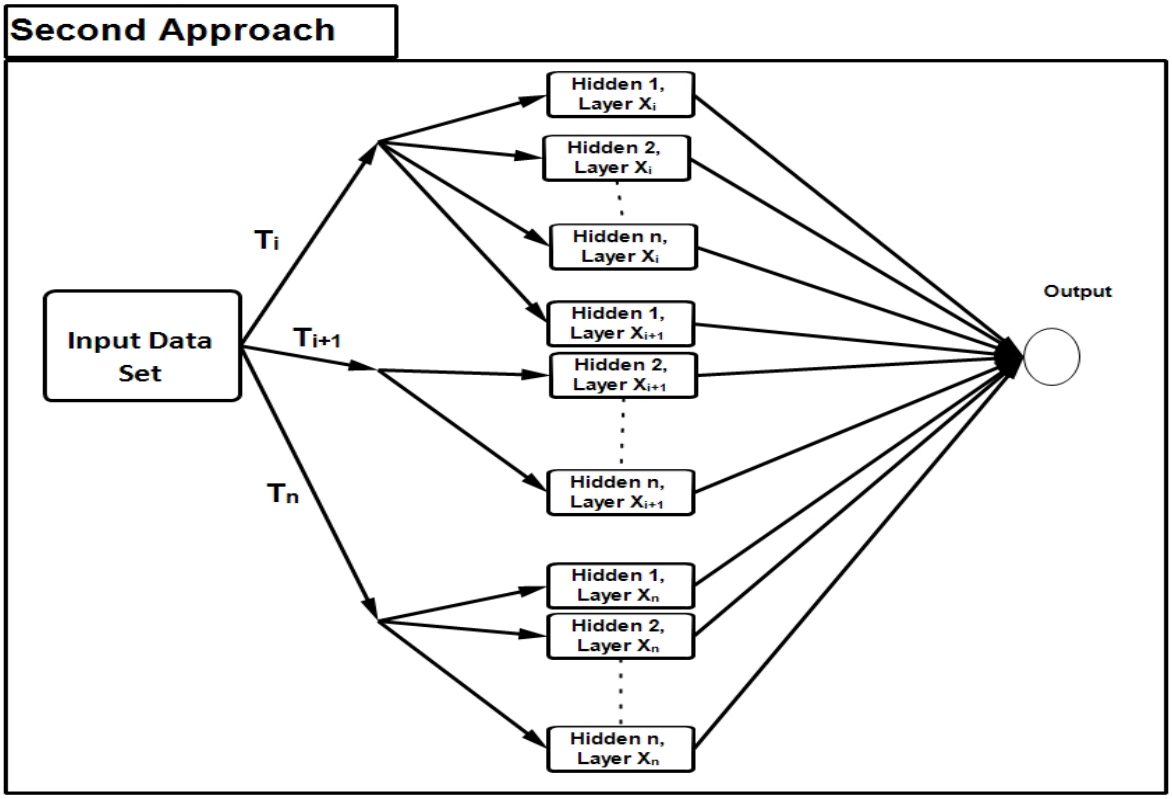

Figure 3. Parallel Mechanism proposed for P-CC-NN using explicit parallelizim model

Computation of the layers tasks of a job with different hidden nodes is performed at the same time. The benefits of performing all layers and hidden nodes together, allow multiple threads to run simultaneously and independently, which minimizes computation time.

3. Once this step of the process is completed the job is then submitted to the queue for inspection. Each thread contains its own *ID* which is attached to it. It is the job of the scheduler to distribute the jobs to the workers that are available to inspect and evaluate.
4. For the final step it is required that the synchronization has to be done before the results can be combined and all the threads eliminated. The reason for this is that the execution times for each hidden nodes and layer have a different value from others; these are established on the training data set. Each thread waits until it is completed and then detaches from the other threads, once this is done it connects the final results that then calculate the accuracy, specificity and sensitivity which was explained earlier. Once the execution of the parallelized code is complete, the threads then join back into the master thread, which continues onwards until the end of the program.

3.2 System Requirements

This section presents the experiments setting of the proposed P-CC-NN algorithm. All experiments were run on JadHPC cluster of FUJITSU PRIMERGY RX 2540 M1, 2 x Intel Xeon E5-2695v3 14C/28T 2.30 GHz, available at the Faculty of Science and Information Technology, Al-Zaytoonah University of Jordan. The operating system is Redhat 7, with the development language being MATLAB 2014b.

3.3 Data Set

In this work, we used a complex case of pattern recognition system which is 3D facial data to examine the proposed system and ensure its effectiveness, in order to ensure that the proposed system can be used in different pattern recognition applications. The dataset used for the experiments was taken from a well-known database called "Face Recognition Grand Challenge (FRGC)", which is

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

considered as the most challenging dataset available for supporting research on 3D face recognition in regard to the expression and pose variations, and the presence of extraneous features [17]. The FRGC database consists of 50,000 2D and 3D images divided into training and validation sets. For this work, 360 3D facial images for 36 persons have been used, each individual represented by 10 images that cover a range of poses and expression, and each image contains 96 distinguishable features. The 3D images were acquired using a Minolta Vivid 900/910 series structured lighting sensor which outputs 640 by 480 range samples and a registered color image with about 160 pixels between the eyes. The Jack-knife technique [18] was employed to evaluate the performances of the learning system used in this work because it has proved to be more appropriate and efficient when applied to different pattern recognition applications. In addition, the use of jack-knife learning algorithms with 80% of the data for training and 20% for testing lead to a reasonably reliable results when it is applied to evaluate a similar classification problem [19].

In this work, 288 images of the available samples were randomly selected and used for training while the remaining 72 images were used for testing. The results were then analyzed to assess the performance.

4. EXPERIMENTS AND RESULTS

This section describes the experiments performed and the results of the experiment on Parallel Cascade Correlation Neural Network (P-CC-NN). The main objective is to enhance the classification phase of any pattern recognition system in a reasonable time. To evaluate the performance of the P-CC-NN, speed up and efficiency are measured. To get the speed up and efficiency of the parallel method, the sequential method was implemented and tested. The first experiment conducted was to examine the first parallel approach of P-CC-NN as in (Figure 2) using MATLAB parfor. Afterwards, an experiment was also conducted to examine the second parallel approach of P-CC-NN as in (Figure 3) using create independent jobs. The number of parallel threads ranged from 1 to 10.

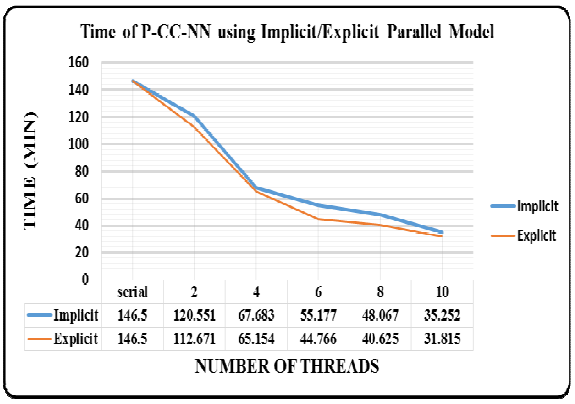

Figure 4. Time of P-CC-NN using Implicit/Explicit Parallel Model

Figure 4 shows the time consumed by P-CC-NN using parfor approach and create independent jobs with varying numbers of threads. The x-axis represents the number of threads whereas the y-axis is the elapsed time in minutes. However, the results show that the processing time required to classification phase of 3D facial recognition system decreased when a number of threads is used in both approaches. For example, the time required, which runs on sequential processor is about 146.5 minutes (2.45 hours). Running the P-CC-NN using implicit model on 10 threads on the other hand takes only about 35 minutes. While, using explicit model it takes about 31 minutes on 10 threads.
Furthermore, it was also observed that P-CC-NN achieves the best time when using explicit parallel model compared to the implicit approach.

Figure 5 shows the speed of P-CC-NN using implicit/explicit parallel model, the x-axis represents the number of threads and the y-axis the speed up. The results show that the implicit approach achieves speedup 1.2 times faster when running on 2 threads than sequential implementation. While the explicit model achieves better speedup when running on 2 compared to implicit model. Furthermore, it was also observed that explicit model achieves the best speed up in general than implicit model.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

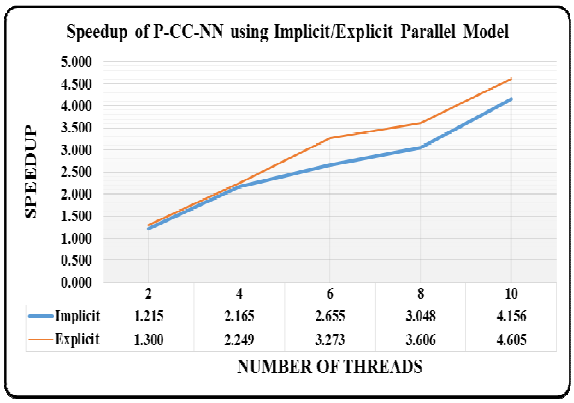

Figure 5. Speedup of P-CC-NN using Implicit/Explicit Parallel Model

Figure 6 shows another aspect of the results obtain from this experiment1 which is the efficiency of P-CC-NN. The x-axis denotes the number of threads and the y-axis the efficiency. The Efficiency (E) is a measure of the fraction of time for which a processing element is usefully employed. It is defined as the ratio of speed up to the number of processing elements. Low efficiency numbers may prompt the user to run the application on fewer threads/processors and free resources to run something else (another threaded process, other users codes etc.). In our case, and even the running time of the P-CC-NN is improved from 146.5 minutes in the case of sequential for 31 minutes in the case of 10 threads, however, the efficiency of P-CC-NN is decreased from 60% in the case of 2 threads to 45% in the 10 threads case.

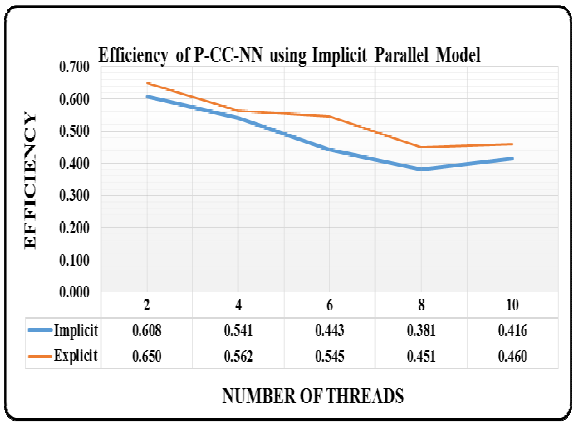

Figure 6. Efficiency of P-CC-NN using Implicit Parallel Model

From the results presented in this section, it was observed that the processing time is better using the second approach of create independent jobs when the number of threads is equal to 10. The second approach took approximately 31.8 minutes when running 10 threads compared with the first approach of MATLAB parfor that took approximately 35.2 minutes when running on 10 threads. This was mostly caused by the communication time incurred between the cores to gather results, or the time when some cores remain idle, or time taken to create new processes or threads. It is also observed that the results of the MATLAB create independent jobs approach (Explicit) are better than the results of MATLAB parfor (Implicit) due to threads' control. In the MATLAB create independent jobs approach, the P-CC-NN controls all the thread's work, which led to less communication time between the cores and the time when some cores remain idle.

5. CONCLUSION AND FUTURE WORK

In this paper, we have proposed multi-core programming model that implements the cascade correlation neural networks technique for 3D facial classification, which had not been done before [2]. In the machine learning stage, the training is carried out using the training vector and the weights of the new hidden nodes are adjusted after each pass. Parallel Cascade Correlation Neural Networks (P-CC-NNs) have a number of attractive features including a very fast training time, often a hundred times faster than other types of networks. The optimization and learning experiments using P-CC-NNs were carried out as explained in the previous section. In order to evaluate the performances of the learning system, the Jack-knife technique was employed with the use of 80% randomly selected samples for training and the remaining 20% for testing. The results indicate that our algorithm achieved significant improvement in execution time about 31 minutes (4.6 times speedup), and generated accuracy value of 94 % with high level of robustness.

Future work to improve the outcome of the current work should include more accurate and efficient techniques for improving the classification phase such as using the multicore programming based on general-purpose computation on graphics processing units (GPGPU) [20]. This algorithm is designed to work in 3D data, another possible

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

improvement is to consider the utilization of this technique to work indifferent biometric systems such as signatures, handwriting, 2D face images and fingerprints.

ACKNOWLEDGEMENT

The authors would like to thank Al-Zaytoonah University of Jordan for their financial support (grant number 2014/25/2).

REFRENCES:

[1] Akarun, L., B. Gökberk, A.A. Salah. 3D Face Recognition for Biometric Applications, in *European Signal Processing Conference*, Antalya, 2005.

[2] Al-Qatawneh, S. and Jaber, K., Parallel Cascade Correlation Neural Network Methods for 3D Facial Recognition: A Preliminary Study. *Journal of Computer and Communications*, 3, 2015, 54-62.

[3] J. Khalid Mohammad, A. Rosni, and R. Nur'Aini Abdul, "Fast decision tree-based method to index large DNA-protein sequence databases using hybrid distributed-shared memory programming model", *Int. J. Bioinformatics Res. Appl.*, January 2014, pp. 321-340.

[4] S. Al-Qatawneh, 3D Facial Feature Extraction and Recognition. *LAP Lambert Academic Publishing*, 2012.

[5] X. Chenghua, W.Y., T. Tieniu, and A. L. Q. Long Quan. Depth vs. intensity: which is more important for face recognition. in *17th International Conference on Pattern Recognition*. 2004.

[6] K. W. Bowyer, K.C., and P. Flynn, A survey of approaches and challenges in 3D and multi-modal 3D + 2D face recognition. *Computer Vision and Image Understanding*, 2006. 101: p. 1-15.

[7] T. Nagamine, T.U., and I. Masuda, 3D facial image analysis for human identification. *International Conference on Pattern Recognition*, 1992: p. 324 - 327.

[8] Fahlmann, S.E., Lebiere, C., Advances in Neural Information Processing System 2(NIPS-2), in *Touretzky*, D.S. 1989: Morgan Kaufmann, Denver. p. 524.

[9] A.A. Huqqani, E. Schikuta, S. Ye, and P. Chen, "Multicore and GPU Parallelization of Neural Networks for Face Recognition", *Procedia Computer Science*, pp. 349-358.

[10] S.-C. Xavier, M.-R. Francisco, and U.-C. Victor, "Parallel Training of a Back-Propagation Neural Network Using CUDA," *Book Parallel Training of a Back-Propagation Neural Network Using CUDA, Series Parallel Training of a Back-Propagation Neural Network Using CUDA*, IEEE Computer Society, 2010,pp.307-312.

[11] M.L. Mark Pethick, Paul Werstein, and Zhiyi Huang, "Parallelization of a Backpropagation Neural Network on a Cluster Computer," *Proc. the Fifteenth IASTED International Conference on Parallel and Distributed Computing and Systems*, November 2003, pp. 574-582.

[12] D. German, " Engineering 90 Project Proposal. Computing Hardware for Accelerated Training of Cascade-Correlation Neural Networks", *Swarthmore College Department of Engineering*, Dec 4, 2007.

[13] L.N. Long, and A. Gupta, "Scalable Massively Parallel Artificial Neural Networks," *Journal of Aerospace Computing, Information*, and Communication, 2015/02/17 2008.

[14] D. George, M. Alan, and N. Tia, "Parallelizing neural network training for cluster systems," *Book Parallelizing neural network training for cluster systems, Series Parallelizing neural network training for cluster systems*, ed., ACTA Press, 2008.

[15] Dobnikar, U. Lotri, B. ter, O. Schuessler, and D. Loyola, "Parallel Training of Artificial Neural Networks Using Multithreaded and Multicore CPUs", *Adaptive and Natural Computing Algorithms, Lecture Notes in Computer Science 6593*, Springer Berlin Heidelberg, pp. 70-79.

[16] K. Ingrid, K. Masafumi, A. Jun-ichi, and T. Hideto, "The time-sliced paradigm—a connectionist method for continuous speech recognition," *Inf. Sci.*1996, pp. 133-158.

[17] Phillips, P.J. FRGC and ICE Workshop–NIST. in *NRECA Conference Facility 2006*. Arlington, Virginia.

[18] Fukunaga, K., Introduction to Statistical Pattern Recognition, in Academic. 1990: New York, USA. p. 220.

[19] Allred, L.G. JackKnife method for validating neural network models in *IJCNN-91-Seattle International Joint Conference 1991*. Seattle, WA.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[20] GPGPU; http://gpgpu.org/, retrieved 2 January 2016.