Journal of Science and Engineering Technology, Vol. 14, No. 1, pp.1-12 (2018)

Design and Implementation of a Visitor Management System by Using Graphics Processing Unit-Accelerated Back-Propagation Neural Networks

CHUAN-WEN CHIANG
Department of Computer and Communication Engineering, National Kaohsiung First University of Science and Technology
No. 1, University Rd., Yanchao Dist., Kaohsiung 82445, Taiwan, R.O.C.
ccw@nkfust.edu.tw

ABSTRACT

This paper presents Bio-IdGate as an advanced visitor management system to improve visitor registration and information management activities. In the Bio-IdGate system, back-propagation neural networks realize personnel recognition schemes to confirm the identity of an individual requesting services. Moreover, the implementation of neural networks by using graphics processing units(GPUs) provides remarkable performance compared with central processing units (CPUs) for computationally-intensive applications, such as training of back-propagation neural networks for large data sets. Experimental results revealed that GPU-accelerated implementation reduced computational costs compared with the standalone CPU version. Bio-IdGate is capable of offering an organized overview of visitor records and reducing time spent managing visitor information.

***Key Words*:** Biometric recognition, back-propagation neural networks, general-purpose computing on graphics processing units (GPGPU), visitor management systems

植基於倒傳遞神經網路與 GPU 加速技術之訪客管理系統設計與實作

江傳文
國立高雄第一科技大學電腦與通訊工程系
82445 高雄市燕巢區大學路 1 號
ccw@nkfust.edu.tw

摘　要

本研究中，吾人設計並實作出一款名為 Bio-IdGate 的自動化訪客管理系統。為了改善傳統作業有關訪客登錄以及訪客資料管理等處理方式的缺失，Bio-IdGate 運用了倒傳遞神經網路技術與 GPU 平行加速運算方法實現其核心功能。在 Bio-IdGate 系統中，根據擷取所得之生物特徵，倒傳遞神經網路被用於實現訪客身份識別之服務。有鑑於倒傳遞神經網路於訓練過程中會隨著資料樣本數量的增加而耗用大量的運算資源，Bio-IdGate 採用自行研發的 GPU 平行

加速運算方法，俾使系統得以提供更為即時的服務。實驗結果顯示，運用 GPU 加速倒傳遞神經網路的執行確實有其成效。整體而言，Bio-IdGate 為傳統訪客管理作業方式的不足提供了有效的解決方案。此一系統預期將可成為企業機構施行安全管制措施所不可或缺的利器。

關鍵詞：生物特徵識別，倒傳遞神經網路，通用目的運算之圖形處理單元，訪客管理系統

I. Introduction

Visitor management is the first line of defense when it comes to the issue of security control in a corporate organization. The primary objectives of visitor management is to identify visitors at the receptionist desk, and to give them sorted authorities which track the identified visitors and limit their activities within some certain areas. Conventionally, visitor management is accomplished manually [1]. Upon arrival at the reception desk, a visitor must present his identification card, and then the receptionist or the visitor fill out visitor information in a logbook. In corporations with greater security needs, the identification card is scanned for the archives, the visitor’s host is contacted, and access controls are issued/returned. Written reports have some obstacles: they are time-consuming and laborious; the handwriting legibility may not be accurate. Moreover, for the relevant information is recorded in written-form, which complicates real-time monitoring procedures and is difficult to inquire when compiling statistics or auditing [13, 17].

In view of these issues, an increasing number of corporations have introduced automated visitor management systems in an attempt to integrate facilities and enhance the effectiveness of their internal security control procedures. With recent advances in information and communications technology, visitor management systems now offer more than just personnel access control but also integrate a wider range of user-friendly services, such as the authority of accessing certain areas or floors, records of visitor activities, compilation of statistics and reports, digital surveillance, and management of parking conditions. This trend has undoubtedly made automated visitor management systems indispensable to corporate security control. The most common automated visitor management systems identify users via keycards. However, in this approach, keycards might be forgotten, lost, or damaged. If a lost keycard is found and used by someone with questionable motives, it may lead to irreparable losses for the corporation. The development of advanced visitor management systems with biometric identification functions has thus received widespread attention.

Biometric identification technologies rely on biological features to differentiate individuals. Essentially, biometric features must possess uniqueness, universality, and permanence [8]. Uniqueness means that the feature must be distinctive; universality means that all individuals must have the same type of feature, and permanence means that the feature does not change with the progress of time (or at least changes very slowly). Furthermore, systems developed from integrative biometric identification technologies must possess collectability, performance, acceptability, and circumvention [10]. Based on these principles, most biometric identification systems utilize facial features, signatures, fingerprints, vein patterns, voices, retinas, or DNA [15-16].

Among these technologies, face recognition is the most natural and undetectable approach. Here, "natural" indicates this biological feature used is same as which humans use to identify individuals. Humans observe and compare facial features to confirm individual identities. Fingerprint and retina recognition in contrast are not natural forms of identification. Undetectability is a crucial characteristic of recognition technologies, as recognition methods which do not attract attention during implementation result in a reduced likelihood of intentional deception. For instance, face recognition relies completely on visible light to obtain facial images and information. In contrast, fingerprints and retinas respectively require electronic pressure sensors and infrared to be collected. These distinct methods of collection are conspicuous and are thus more likely to be deceived. For this reason, this study developed Bio-IdGate, which is an automated visitor management system with biometric identification functions. Bio-IdGate achieves visitor identification via facial images or electrocardiograms (ECGs) from wearable devices. With this system, corporations can integrate their relevant facilities and further enhance the effectiveness of internal security procedures. To provide more prompt services, Bio-IdGate uses a parallel computation approach that we developed to fulfill the computational needs of the system during its execution. On the whole, Bio-IdGate offers effective solutions to the shortcomings of conventional visitor management procedures. It is predicted that Bio-IdGate will be an essential element of corporate security measures.

The remainder of this paper is organized as follows. Section 2 introduces the relevant technologies involved in the development process of the Bio-IdGate system. Next, Section 3 explains the design methodology of the Bio-IdGate system in detail, and Section 4 presents the implementation results. Section 5 contains experiment results, which we used to evaluate the performance and characteristics of the proposed algorithm. Finally, Section 6 outlines the conclusions of this study.

II. Technology Review

1. OpenCV

OpenCV (Open Source Computer Vision) is a cross-platform computer vision library that facilitates the construction of an easy-to-use computer vision framework so that developers can design more complex applications [6]. This library encompasses numerous computer vision applications, such as medical imaging, user interfaces, three-dimensional vision, biometrics, and robotics. In the Bio-IdGate system, this library realizes functions such as facial detection and feature extraction.

Detecting accurately human faces in complex backgrounds is the first step of face recognition. Fundamentally, the methods can be (1) feature-based or (2) learning-based. The former uses the distribution characteristics of skin color, face shapes, eyes, mouths, and noses to detect faces. However, interferences from the background often affect how facial features are presented, so this processing approach is generally paired with other mechanisms. In contrast, learning-based methods first teach the classifiers to learn from the different distribution characteristics of faces, then the classifiers analyze all the images to locate human faces. This approach can detect human faces even when facial features are not clear. For the sake of accuracy, Bio-IdGate integrated a learning-based approach with a Haar feature-based cascade classifier to detect human faces in the Bio-IdGate system. By means of Haar-like features (represented in Fig.1) obtained from the quick computation of integral images [20]. For input image *I*, the integral image value of the pixel at (x, y) is defined as

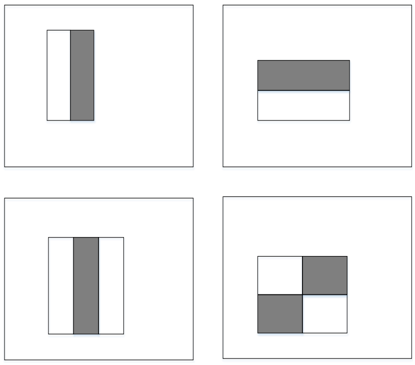


Fig. 1. Haar-like features.

$$ii(x, y) = \sum_{x' \le x,\, y' \le y} i(x', y') \tag{1}$$

where $i(x', y')$ denotes the value in the original image. Thus, obtaining the integral image of image *I* requires pointwise scanning to calculate the following:

$$s(x, y) = s(x, y\text{-}1) + i(x, y) \tag{2}$$

$$ii(x, y) = ii(x\text{-}1, y) + s(x, y). \tag{3}$$

2. Artificial neural networks

Once the biometrics have been detected and extracted, the Bio-IdGate system uses an artificial neural network (ANN) to distinguish the identification. ANN is a computational model that use numerous connecting artificial neurons to imitate the biological neural networks. Artificial neurons are simple simulations of biological neurons. As the basic units of ANNs, neurons are in charge of generating, transmitting, and processing signals in the neural network. Via their dendrites, nerve cells receive signals from other nerve cells and then process the signals in their nucleus. The processing involves summing the information collected and then performing a nonlinear conversion to create a new signal. Neurons have a threshold for signal strength; if the strength of a signal exceeds that threshold, then the new signal will be transmitted to other nerve cells via the axon.

The more common used ANNs for classification or prediction involve back-propagation [3-4, 7, 14], radial basis functions [11, 19], and wavelets [2], etc.. Among which back-propagation is the most widely used. Back-propagation neural networks consist of multilayer feedforward neural networks which is able to process self-learning mechanism.

Aside from the input and output layers, back-propagation neural networks also have at least one hidden layer. Fig. 2 illustrates a back-propagation neural network with a three-layer structure [5]. In back-propagation neural networks, neurons on the same layer are isolated from each other; however, for those neurons on adjacent layers, they are connected with each other. Neurons on the input layer responsible for receiving input signals from the outside world (such as facial features) and then sending the signals to neurons on the hidden layer. The primary function of neurons on the hidden layer is to first present the results of the interactions among the neurons on the previous layer using weight values, then to identify the internal structures and characteristics of the problem being processed. Neurons on the output layer receive the signals transmitted from the neurons on the hidden layer and then send out the final computation results (such as identities).

Fundamentally speaking, the learning process of a back-propagation neural network can be regarded as a supervised learning method. With the neural network in Fig. 2 as an example, the details of this learning process are as follows.

Step 1: Define the structure of the neural network.

This step determines the number of layers in the structure of the network, and the number of neurons on each layer as well. In Fig. 2, we assumed that the numbers of neurons on the input layer, hidden layer, and output layer are *m*, *l*, and *n*, respectively.

Step 2: Initialize the biases of each neuron and the weights among them.

Let N_i ($1 \le i \le m$), N_j ($1 \le j \le l$), and N_k ($1 \le k \le n$) represent any given neuron on the input layer, hidden layer, and output layer, respectively. Randomly determine the values of θ_j, θ_k, w_{ij}, and w_{jk}, where θ_j is the bias of neuron N_j; θ_k is the bias of neuron N_k; w_{ij} is the weight between neurons N_i and N_j, and w_{jk} is the weight between neurons N_j and N_k. It must be noted that neurons on the input layer only send the signals that they receive to neurons on the hidden layer. They do not participate in the computation. In other words, only the biases of the neurons on the hidden layer and the output layer need to be determined.

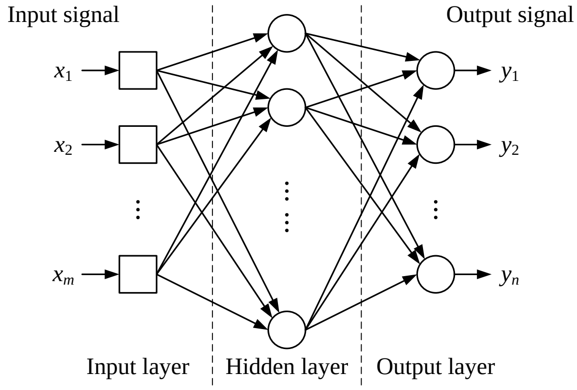


Fig. 2. A feedforward network with one hidden layer.

Step 3: Load training data.

The data required for this step include input vector $\boldsymbol{X} = (x_1, x_2, ..., x_m)$ and the expected output vector $\boldsymbol{D} = (d_1, d_2, ..., d_n)$.

Step 4: Calculate inferred output vector $\boldsymbol{Y} = (y_1, y_2, ..., y_n)$.

Prior to calculating the inferred output vector $\boldsymbol{Y}$, we must obtain the contents of $\boldsymbol{H} = (h_1, h_2, ..., h_l)$, the vector output by the hidden layer. The value of any element h_j within output vector $\boldsymbol{H}$ can be written as

$$h_j = \frac{1}{1+e^{-net_j}} \tag{4}$$

where net_j denotes the weighted product of hidden layer neuron N_j and is calculated using the formula below:

$$net_j = \sum_{i=1}^{m} x_i \times w_{ij} - \theta_j \tag{5}$$

After obtaining the content of output vector $\boldsymbol{H}$, we can infer the value of any element y_k in output vector $\boldsymbol{Y}$ using the same method. That is,

$$y_k = \frac{1}{1+e^{-net_k}} \tag{6}$$

where

$$net_k = \sum_{j=1}^{l} h_j \times w_{jk} - \theta_k \tag{7}$$

Step 5: Calculate the error gradient.

Let δ_j and δ_k be the error gradients of hidden layer neuron N_j and output layer neuron N_k. We can first calculate δ_k using the formula below:

$$\delta_k = y_k \times (1 - y_k) \times (d_k - y_k)\,. \tag{8}$$

Once the error gradients of all the neurons on the output layer have been calculated, we can use the formula below to derive the error gradient δ_j of any neuron N_j on the hidden layer:

$$\delta_j = y_j \times (1 - y_j) \times \sum_{k=1}^{n} \delta_k \times w_{jk}\,. \tag{9}$$

Step 6: Update the biases of each neuron and the weights among them.

Like the calculation of the error gradients, the update operation is conducted backwards as well. First, the weight between hidden layer neuron N_j and output layer neuron N_k is updated as follows:

$$w_{jk} = w_{jk} + \Delta w_{jk} \tag{10}$$

where

$$\Delta w_{jk} = \eta \times h_j \times \delta_k \tag{11}$$

In the formula above, η denotes a learning rate that is generally set between 0.1 and 1.0. Next, the bias of output layer neuron N_k is updated as follows:

$$\theta_k = \theta_k + \Delta\theta_k \tag{12}$$

where

$$\Delta\theta_k = -\eta \times \delta_k \tag{13}$$

Once the biases of all of the output layer neurons and their weights with the neurons in the hidden layers have been updated, we can update the biases of the hidden layer neurons and their weights with the neurons in the output layers.

Step 7: Perform convergence test.

If the results of the convergence test indicate that further training is needed, the process goes back to Step 3, redo the following calculations. If not needed, then the training is completed.

Once the neural network has been trained, we can solve problems related to classification, prediction, and noise filtering in the subsequent recall phase. The procedure of this phase is as follows:

Step 1: Define the structure of the neural network.

Step 2: Use training results to set the biases of each trained neuron and the weights among them.

Step 3: Load test vector ***X***.

Step 4: Calculate the inferred output vector ***Y***.

Back-propagation neural networks offer high learning accuracy and recall speed. However, a greater number of training samples results in more repetitions of the learning process, which reduces the learning speed. Furthermore, the initial weights of the neurons, and the input threshold value and learning rate all exert influence on the speed of the training. The numbers of hidden layers and neurons also impact learning speed and recall accuracy. In response, researchers have proposed the use of computational intelligence techniques (such as particle swarm optimization) to optimize back-propagation neural networks [21].

3. OpenCL

During the training phase, the computational costs of back-propagation neural networks increase exponentially with the number of training samples. One can reduce the training time by selecting initial values [12], controlling the learning parameters [9] and determining weight adjustments [22]. However, most realistic approaches still remain to be improved. We therefore used OpenCL [18] to incorporate a novel parallel computation method into the Bio-IdGate system and enhance the execution performance and service quality of the system. OpenCL is an open standard that can be considered as a framework, in which, one can run parallel programs on a distributed heterogeneous system. OpenCL provides parallel computation mechanisms that aim on task and data partitioning.

Any programs with parallelizable computation processes can be accelerated using OpenCL, and so far, it has been applied to weather forecasting, linear algebra, and distributed searches. The computational characteristics of the equipment being used can also influence the effectiveness of parallel computation with OpenCL. GPUs, for example, are designed to process large amounts of simple computations. If the parallelized portions of programs suit GPUs' characteristics as mentioned, simple and large, then parallelization will produce good results.

In Bio-IdGate, OpenCV is applied to facial detection and feature extraction. ANNs and OpenCL are also utilized to make identification and to accelerate the computation of the ANN training phase, respectively. The integration of OpenCV, ANNs and OpenCL can reduce implementation complexity for creating a visitor management system.

III. Design Methodology

This section discusses a few design-related issues in Bio-IdGate, including (1) system requirements for Bio-IdGate, (2) the way the biometric identification engine works, and (3) parallel computation with GPUs. We first present requirements for developing the Bio-IdGate system from a dynamic and static perspective respectively. We then explain how the back-propagation neural network is incorporated to realize the biometric identification engine function. Finally, we describe in detail how GPUs can be used to implement parallel computation and enhance system performance.

1. System requirements

In this study, we divided the visitor management process into three phases: (1) preparation, (2) reception, and (3) departure. During the preparation phase, visitors plan their visit by making an appointment with their host. If the visit is approved by a supervisor, the visitor's basic information and the time of the appointment are entered into the visitor management system to create a visit form. When users enter the name of the visitor in the creation of the visit form, the system will automatically check whether the visitor has visited before. If relevant records exist, the personal information of the visitor (such as ID number, name, affiliation, and contact number) is loaded into a graphical user interface, thereby simplifying the visit form creation process. In the event that a visitor does not make an appointment or receive approval, the visitor can apply at the reception desk.

When the visitor arrives at the corporate organization at the appointed time, the reception phase begins. If a visit form has already been created for the visitor in the visitor management system, the visitor need only sign in and submit his ID to receive an access card. The sign-in process can be completed via the biometric identification function. Visitors without appointments can apply at the reception desk, which will first contact the visitor's host. Following verification and approval, the receptionist assists the visitor in creating a visit form, signing in, and then exchanging his ID for an access card. Finally, when the visit ends, the visitor must return his access card to the reception desk, which constitutes the departure phase.

Based on the description of the context above, the use case diagram associated with the functions of the system is as shown in Fig.3. The actors of the Bio-IdGate system include receptionists, hosts, and managers, and the services provided by the system include visitor information management, visit form management, user account management, and report management. The functions for visitor information management, visit form management, and user account management enable users to add, edit, delete, and query information.

Furthermore, the static data perspective in the Bio-IdGate system includes the following:

- Each visitor may have multiple (or no) visit forms; each visit form may only pertain to one visitor.
- Each visitor may have multiple (or no) photos in the system; each photo may only pertain to one visitor.
- Each visit form must be under the charge of only one employee (the host); each employee may be in charge of multiple (or no) visit forms.
- Each visit form may have one access card attached; each access card may only correspond to one visit form.

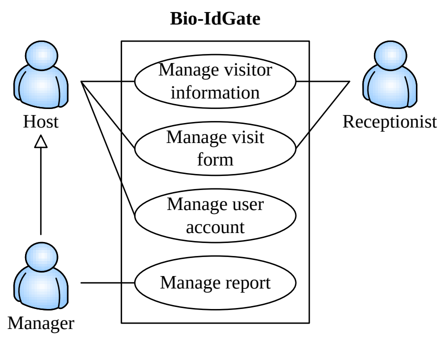


Fig. 3. Use case diagram of the Bio-IdGate system.

2. Operation of the biometric identification engine

The core of the Bio-IdGate system is the means of accurately identifying visitors. The developed system employs back-propagation neural networks and OpenCL acceleration to produce better classification accuracy and performance than existing systems. Similar to the operation process of a back-propagation neural network, the implementation of the biometric identification engine of Bio-IdGate includes a learning phase and a recall phase. Every time new visitor information is added, the learning phase is re-run. During this phase, biometric feature values extracted from images are input into the input layer and processed by the hidden layer, after which the results are output by the output layer. Let d_{sk} and y_{sk} be the target and actual output values of a sample s. Using the error function below, we can calculate the discrepancy between the two:

$$E_s = \sum_{k=1}^{n} (d_{sk} - y_{sk})^2 \ . \tag{14}$$

In Bio-IdGate, moreover, the sum of squared error (SSE) is used as the termination criterion. Suppose that there are $|S|$ samples, the SSE between network outputs and desired targets is as follows:

$$SSE = \sum_{s=1}^{|S|} E_s \ . \tag{15}$$

3. GPU parallel computation

Back-propagation neural networks are a type of artificial neural network that simulates the transmission of electric signals among neurons in the human brain to exhibit learning and reasoning abilities. In the human brain, nerve cells are intertwined and pass on information to one another. In a single-threaded environment, simulating this process requires dealing with the signal transmissions among neurons one step at a time. Increasing the number of neurons means increasing the computation time significantly, and this problem is particularly severe in the learning phase. Fortunately, this process can be parallelized in a multithreaded environment. Therefore, the Bio-IdGate system uses OpenCL and the hundreds and thousands of computing units on GPUs to parallelize and accelerate the learning process of the neural network.

In the sequential processing of array data, one or more layers of for-loops can be used to access each element in the array, depending on the number of dimensions. Taking a two-dimensional $m \times n$ matrix M as an example, we present the program segment below in which each element in the array is processed once:

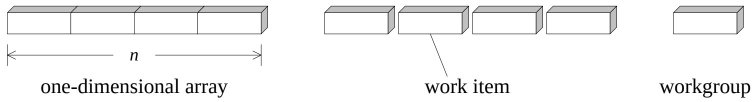
```
for(i=0;i<m;i++)
    for(j=0;j<n;j++)
        process(M[i][j]);
```

The processing of each element can be regarded as a work item, and a workgroup comprises at least one work item. Through OpenCL, each workgroup is assigned to a computing unit for computation. The method of partitioning the data into workgroups comprising individual work items and making reasonable use of the computing units in the GPU to process different parts of the same set of data at the same time are the linchpins of program implementation.

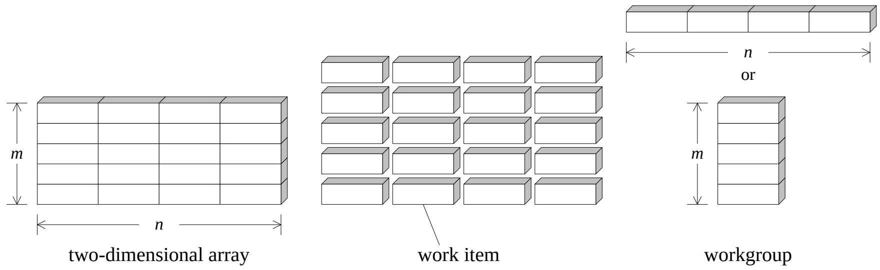
Generally speaking, data partitioning involves the means of dividing the data into work items and then forming workgroups. In the Bio-IdGate system, there are three partitioning strategies:

- Divide a one-dimensional array with length n into n work items, each of which constitute a workgroup (as shown in Fig. 4(a)), such as the bias and error gradients of the neurons in the hidden and output layers.
- Divide a two-dimensional $m \times n$ array into $m \times n$ work items with m or n work items forming a workgroup (as shown in Fig. 4(b)). This task is determined by the number of neurons in each layer, and the principle "the more workgroups the better" determines how many work items each workgroup contains. Examples include the weights between the neurons of the output layer and the hidden layer and between the neurons of the hidden layer and the output layer.
- Divide a two-dimensional $m \times n$ array into m or n work items with each work item forming a workgroup (as shown in Fig. 4(c)). Examples include the weight arrays between neurons when calculating the output signals of the nodes of the hidden and output layers.

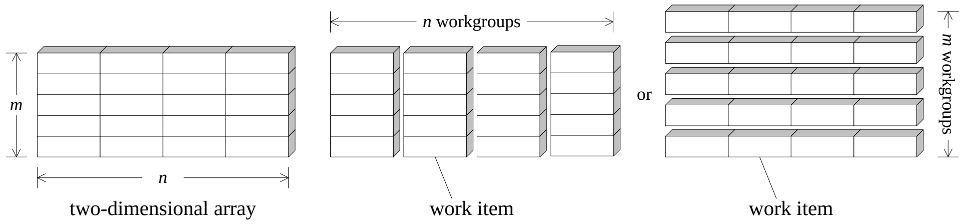
It must be noted that dividing the data into a greater number of work items leads to a greater number of workgroups. However, when the number of workgroups exceeds the number of computing units in the GPU, the workgroups must be given to the computing units in batches for processing. A greater number of workgroups therefore means a longer computation



(a) Data partitioning strategy #1



(b) Data partitioning strategy #2



(c) Data partitioning strategy #3

Fig. 4. Data partitioning strategies used in Bio-IdGate.

time. Partitioning data using two-dimensional matrices and designating single-element workgroups creates an excessive number of workgroups as the number of nodes increases, which in turn brings about the aforementioned problem. The second approach for data partitioning overcomes this by increasing the number of workgroups in a single layer as the number of nodes increases rather than using the product of the numbers of nodes in two layers.

Furthermore, whether the computations in a back-propagation neural network can be parallelized is also determined by whether the input and output data have a precedence relationship and whether they influence each other during access. When calculating the output signals of the neurons in the hidden and output layers, partitioning each weight into a workgroup, multiplying them by the input signals, and then adding them to the corresponding neurons results in multiple computing units revising the output value of a single neuron. As the command used for the summing operation is not atomic, this presents the problem of competition. To resolve this issue, we employed the third approach of data partitioning and handed the computation of single node output values over to single computing units.

IV. Implementation

The Bio-IdGate system currently offers multiple methods of identification: face recognition, QR code identification, and ECG identification. The face recognition function makes use of system cameras to detect and identify a human face and then presents the personal data and visit form corresponding to the visitor (represented in Fig.5). Visitor identification can also be made using QR codes; if successful, the system will present the personal data and visit form corresponding to the visitor, but if not, the visitor data field holds a question mark. As shown in Fig.6, users can add, edit, delete, and query information regarding visitors and visits. When users click the function buttons, the corresponding window pops up. Clicking the "Sign In" button on the visit form will change the visitor’s status from "Not signed in" to "Signed in".

Fig. 5. Screenshot of visitor registration.

Fig. 7. Screenshot of visitor status monitoring.

Fig. 6. Screenshot of visitor information editing.

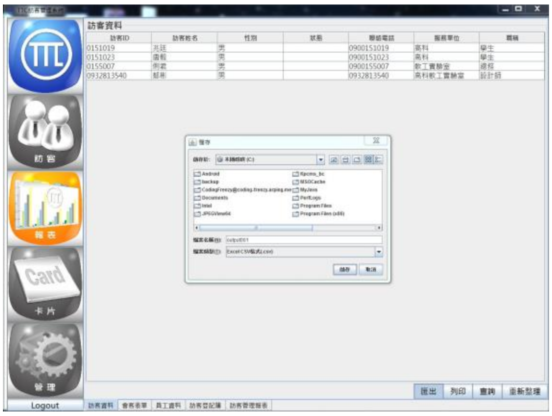


Fig. 8. Screenshot of report generating.

Users can also look up the information of visitors who have signed in and not yet left in the View tab of the Visitor section (represented in Fig.7). By clicking the Sign Out button for a visitor, users can sign visitors out, following which the status of the visit form will change from "Signed in" to "Signed out". In the Report section, the system lists information regarding visitors and visits in a table. As shown in Fig.8, Users can select or search for information based on conditions and output the information they require in one of three formats. In the Visitor Information and Contact Information tabs of the Management section, users can add, edit, delete, and query information. Fig.9 displays the system setting tab of the Management section, where three types of system settings can be adjusted: the OpenCL-accelerated back-propagation neural network, the original back-propagation neural network, and the PSO-optimized back-propagation neural network. After inputting parameter settings, users can click the "Train" button to train the classifier.

Fig. 9. Screenshot of performance tuning.

V. Performance Evaluation

The purpose of this section is to demonstrate the significance of the proposed GPU parallel computation approach to the enhancement of the performance of the Bio-IdGate system. We used the C programming language to implement our algorithm. All experiments were performed on an Intel personal computer equipped with an NVIDIA GeForce GTX 650 graphics card, 2.94.0 GHz CPU, 4 GB memory, and Microsoft Windows 7 Ultimate SP1 as the operating system.

We assumed that the Bio-IdGate system contained image data of N visitors (each with only one photo) to train the back-propagation neural network classifier. In this classifier, the respective number of nodes both in the input and output layers of the classifier was both set as N, whereas the number of nodes in the hidden layer was set as $2N$. The number of iterations in the training phase was fixed at 200. Fig.10 presents the differences in time costs of computing various data quantities with and without the GPU parallel computation approach, and Fig.11 displays the speedup ratios.

The contents of Fig.10 illustrate the following:

- Without the GPU parallel computation approach, the time costs of the back-propagation neural network classifier increase exponentially with the scale of the image data.
- With the GPU parallel computation approach, the effects of the Bio-IdGate system become more prominent as the scale of the image data increases.

Essentially, CPUs can make complex logical decisions, loops, and control process branching, whereas GPUs specialize itself in performing large amounts of simple and repetitive calculations. As the GPU was employed to train the back-propagation neural network, an enlarging scale of image data increases the number of nodes in the network; the number of nods in turn influences the number of computing units used. It means that more computing units are simultaneously employed. As shown in Fig.11, the speedup ratio of the GPU with 50 images is approximately 0.5. At this point, the accelerating effects of GPU parallelization are not yet apparent because processing the workgroups with the GPU computing units incurs certain communication time costs. However, when the number of images exceeds 80, the effects of parallelization begin to appear, and the speedup ratio of the GPU surpasses 1.

We also observed how the number of work items in each workgroup influences the system performance. With a fixed total image quantity of 200, we measured the training time required for workgroups comprising 1, 5, 10, 20, and 40 work items, respectively, the results of which are as shown in Fig. 12. With one work item comprising each workgroup, the time needed to execute the training loops was approximately 39.268 sec, whereas with 10 work items constituting each workgroup, the time decreased to 13.885 sec. It indicates that workgroups

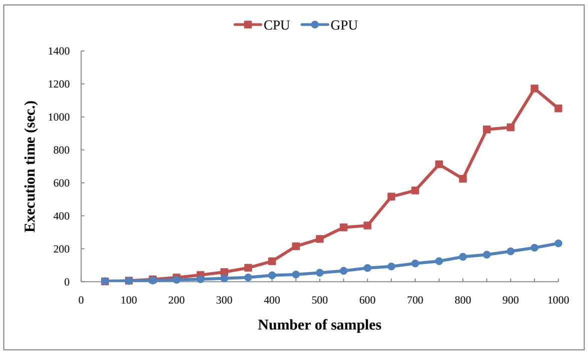


Fig. 10. Influence of GPU parallel computation approach on Bio-IdGate system performance.

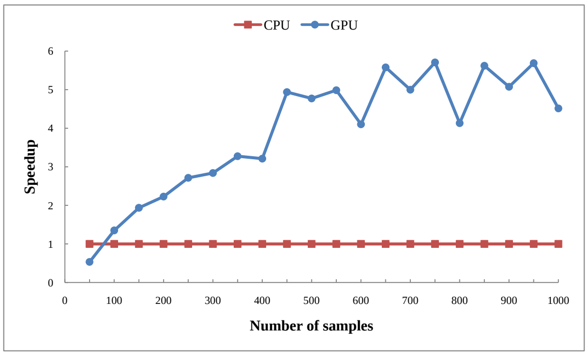


Fig. 11. Speedup ratios of CPU and GPU.

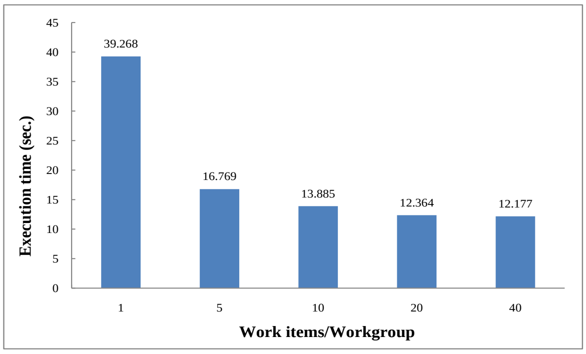


Fig. 12. Influence of the number of work items in each workgroup on Bio-IdGate system performance.

should contain more work items so as to decrease the number of workgroups and reduce the time required to process the workgroups in batches.

The proposed Bio-IdGate can reach the performance of the accuracy to 94% while used in practice. However it is necessary to be aware that the aim of this study is to employ GPU in speeding a visitor management system carries a biometric recognition. Therefore, in spite of numerous approaches that can improve accuracy rate [21], yet the detailed discussion of this issue is outside the scope of this paper.

VI. Conclusion

This study designed and implemented a visitor management system called Bio-IdGate. The system integrates OpenCV, a back-propagation neural network, and OpenCL to provide corporate organizations with comprehensive biometric identification functions. On the whole, the Bio-IdGate system features the following:

- biometric identification, which offers a complete solution to the inadequacies of conventional visitor management procedures;
- a novel GPU parallel computation approach, which provides more instantaneous service;
- a user-friendly graphical user interface that effectively reduces costs for personnel training.

With rapid developments in technologies associated with the Internet of Things (IoT), the Bio-IdGate system can be further developed into a data integration platform that provides indispensable applications for smart living. The Bio-IdGate system can also be further improved with a wider range of biometric identification functions and further increases in the efficiency of the GPU parallel computation method. Incorporating the functionality of iris identification into Bio-IdGate is also considered in the future.

References

1. Anwar, N., M. N. Masrek and Y. R. Rambli (2012) Visitor management system by applying the model of UTAUT. Proceedings of IEEE Symposium on Business, Engineering and Industrial Applications, Bandung, Indonesia.
2. Avdakovic, S., A. Nuhanovic, M. Kusljugic and E. Becirovic (2014) Applications of wavelets and neural networks for classification of power system dynamics events. *Turkish Journal of Electrical Engineering and Computer Sciences*, 22(2), 327-340.
3. Cerqueira, J. J. F., A. G. B. Palhares and M. K. Madrid (2002) A simple adaptive back-propagation algorithm for multilayered feedforward perceptrons. Proceedings of IEEE International Conference on Systems, Man and Cybernetics, Yasmine Hammamet, Tunisia.
4. Chen, D. S. and R. C. Jain (1994) A robust back-propagation algorithm for function approximation. *IEEE Transactions on Neural Networks*, 5(3), 467–479.
5. Chen, J. F., Q. H. Do and H. N. Hsieh (2015) Training artificial neural networks by a hybrid PSO-CS algorithm. *Algorithms*, 8(2), 292-308.
6. Culjak, I., D. Abram, T. Pribanic, H. Dzapo and M. Cifrek (2012) A brief introduction to OpenCV. Proceedings of 35th International Convention on Information and Communication Technology, Electronics and Microelectronics, Opatija, Croatia.
7. Gori, M., and A. Tesi (1992) On the problem of local minima in back-propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1), 76–86.
8. Jain, A. K., A. Ross and S. Prabhakar (2004) An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1), 4-20.
9. Kanan, H., and M. Khanian (2012) Reduction of neural network training time using an adaptive fuzzy approach in real time applications. *International Journal of Information and Electronics Engineering*, 2(3), 470-474.
10. Kumar, D., and Y. Ryu (2008) A brief introduction of biometrics and fingerprint payment technology. Proceedings of Second International Conference on Future Generation Communication and Networking Symposia, Sanya, Hainan Island, China.
11. Lin, C. J. and C. C. Peng (2011) A self-adaptive quantum radial basis function network for classification applications. *International Journal of Innovative Computing, Information and Control*, 7(12), 6621-6634.
12. Nguyen, D. and B. Widrow (1990) Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. Proceedings of International Joint Conference on Neural Networks, San Diego, CA, USA.
13. Noorhuzaimi, M. N., S. Junaida, A. Noraziah and K. Chen (2008) E-visitor information management system (E-VIMS) using MyKad. Proceedings of First International Conference on the Applications of Digital Information and Web Technologies, Ostrava, Czech Republic.
14. Paliwal, M., and U. A. Kumar (2009) Neural networks and statistical techniques: A review of applications. *Expert Systems with Applications*, 36(1), 2–17.
15. Prabhakar, S., S. Pankanti and A. K. Jain (2003) Biometric recognition: security and privacy concerns. *IEEE Security and Privacy*, 1(2), 33-42.
16. Rodrigo, L. G., A. L. Carlos, A. Otman and R. A. Juan

(2003) Biometric identification systems. *Signal Processing*, 83(12), 2539-2557.

17. Satari, B. S., N. A. Abd Rahman and Z. M. Zainal Abidin (2014) Face recognition for security efficiency in managing and monitoring visitors of an organization. Proceedings of IEEE Symposium on Biometrics and Security Technologies, Kuala Lumpur, Malaysia.
18. Scarpino, M. (2012) *OpenCL in Action: How to Accelerate Graphics and Computation*, Manning, Shelter Island, N.Y.
19. Ture, M., I. Kurta, A. T. Kurumb and K. Ozdamar (2005) Comparing classification techniques for predicting essential hypertension. *Expert Systems with Applications*, 29(3), 583–588.
20. Viola, P. and M. Jones (2001) Rapid object detection using a boosted cascade of simple features. Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, USA.
21. Wang, C., Z. Shi and F. Wu (2017) An improved particle swarm optimization-based feed-forward neural network combined with RFID sensors to indoor localization. *Information*, 8(9), 1-18.
22. Wu, P., S. C. Fang and H. Nuttle (1999) Curved search algorithm for neural network learning. Proceedings of International Joint Conference on Neural Networks, Washington, DC, USA.

收件：**106.07.21**　修正：**106.09.26**　接受：**106.11.09**