High Performance Personalized Heartbeat Classification Model for Long-Term ECG Signal

Pengfei Li, Yu Wang, Jiangchun He, Lihua Wang, Yu Tian, Tian-shu Zhou, Tianchang Li and Jing-song Li*

Abstract—**Long-term ECG has become one of the important diagnostic assist methods in clinical cardiovascular domain. Long-term ECG is primarily used for the detection of various cardiovascular diseases that are caused by various cardiac arrhythmia such as myocardial infarction, cardiomyopathy, and myocarditis. In the past few years, the development of an automatic heartbeat classification method has been a challenge. With the accumulation of medical data, personalized heartbeat classification of a patient has become possible. For the long-term data accumulation method, such as the holter, it is difficult to obtain the analysis results in a short time using the original method of serial design. The pressure to develop a personalized automatic classification model is high. To solve these challenges, this paper implemented a parallel general regression neural network (GRNN) to classify the heartbeat, and achieved a 95% accuracy according to the Association for the Advancement of Medical Instrumentation (AAMI). We designed an online learning program to form a personalized classification model for patients. The achieved accuracy of the model is 88% compared to the specific ECG data of the patients. The efficiency of the parallel GRNN with GTX780Ti can improve by 450 times.**

Index Terms—**Classification, generalized regression neural network (GRNN), graphics processor unit (GPU), long-term ECG.**

I. Introduction

Cardiac arrhythmia is an important group of cardiovascular diseases. The disease can either suddenly lead to death or gradually lead to heart failure. Most diagnoses of cardiac arrhythmia depend on the electrocardiogram (ECG). However, some arrhythmias appear infrequently, and prolonged ECG recordings are needed to capture them. In 1957, Dynamic Electrocardiography (DCG, holter) was developed for a long duration monitoring of changes in electrocardiography. Compared to a normal ECG, holter was used to monitor and record a long-term electrocardiography. The data, which is recorded by a holter, may be accumulated for 24 hours or longer and contain hundreds of thousands of heartbeats. There is a higher detection rate for non-persistent arrhythmia, transient arrhythmia and transient myocardial ischemia. Long-term ECG has become an important diagnostic method in clinical cardiovascular domain. It is primarily used in the detection of various cardiovascular diseases that are caused by various cardiac arrhythmia such as myocardial infarction, cardiomyopathy, and myocarditis. The first step in ECG analysis is the segmentation of heartbeats. The second step is the classification of heartbeats according to heartbeat features [1]. However, an artificial analysis of ECG is time consuming. Thus, researchers developed an automatic heartbeat classification method [2].

Automatic heartbeat classification includes elimination of baseline drift [3-7], waveform detection [8-10], feature extraction [11], and heartbeat classification [12-21]. Heartbeat classification is at the core of the automatic ECG analysis. In the past few years, many researchers proposed different heartbeat classification techniques. Some groups used a waveform feature, such as [12-15], and others used a wavelet transform, such as [16, 17]. In recent years, with the development of machine learning techniques, more studies have been conducted on the automatic heartbeat record classification methods to improve the effectiveness of arrhythmia detection. Specifically, [18] used SOMNN (self-organization map neural network), [19] used MLPNN (Multilayer Perceptron Neural Network), [20] used PNN (Probabilistic Neural Network) and [21] used Dynamic Bayesian network.

Although there is much research on the classification of heartbeats, and many high precision classification models have been developed by the researchers, two problems remain. First, the current heartbeat classification models are fixed and are based on a certain sample base training. In practical application, when evaluating patients with large differences of training samples, the fixed models do not perform well as those in the experiment. Because the heartbeat of each patient has its own

This work was supported by the National Natural Science Foundation (Grant No.61173127), National High-tech R and D Program (No.2013AA041201 and No.2015AA020109) and Zhejiang University Top Disciplinary Partnership Program.

Pengfei Li, Yu Tian, Yu Wang,Tian-shu Zhou authors are with the Engineering Research Center of EMR and Intelligent Expert System, Ministry of Education, Collaborative Innovation Center for Diagnosis and Treatment of Infectious Diseases, College of Biomedical Engineering and Instrument Science, Zhejiang University,Hangzhou,China(e-mail:21115055@zju.edu.cn;ty.1987823@163.com; daisy222542@zju.edu.cn; alexgzhou@163.com).

Jiangchun He, Lihua Wang, Tianchang Li are with Heart Center, Navy General Hospital, Beijing, China (e-mail: h2005helen@yeah.net; wanglihua66666@163.com; ltc909@aliyun.com).

*Jing-song Li is with the Engineering Research Center of EMR and Intelligent Expert System, Ministry of Education, Collaborative Innovation Center for Diagnosis and Treatment of Infectious Diseases, College of Biomedical Engineering and Instrument Science, Zhejiang University, Hangzhou, China; (e-mail: ljs@ zju.edu.cn).

characteristics, a machine-learning model is needed with the ability of online-learning. With the accumulation of medical data, a personalized medicine has become possible. Starting with a fixed model, a personalized automatic classification model can be slowly constructed using the heartbeat characteristics of each patient. Second, with the wide use of long-term ECG and the improvements in sampling rate of ECG acquisition devices, a large amount of data is accumulated. It is difficult to obtain analysis results in a short period of time using the original serial design method. The pressure to develop a personalized automatic classification model is high.

To solve the problems of heartbeat classification, this study designed an effective method that is based on GPU, which includes a parallel method for extracting the features of heartbeats and a parallel heartbeat classification method. The research in this paper primarily consists of three parts. 1) Separation of a continuous ECG signal into heartbeats, and the extraction of features of each beat. 2) Design of a parallel GRNN (generalized regression neural network). According to AAMI, we classify five heartbeat types: N (beats originating in the sinus mode), S (supraventricular ectopic beats), V (ventricular ectopic beats), F (fusion beats), and Q (unclassifiable beats). For the two heartbeat types (VEB, SVEB), a binary classification is made. 3) Verification of the model accuracy using the data from real patients. Then, we updated the fixed model into the personalized automatic classification model.

The structure of this paper is as follows. In the second part of the paper, we introduce a parallel method for feature extraction and the classification model. The third part includes the classification of results in our experiments. The fourth part discusses the results after base classification and the results after online learning. In this part, we compared our results with the recent research and displayed the efficiency and advantages of our design. In the fifth part of this paper, we conclude the study.

II. Method

A. Parallel Programming Model

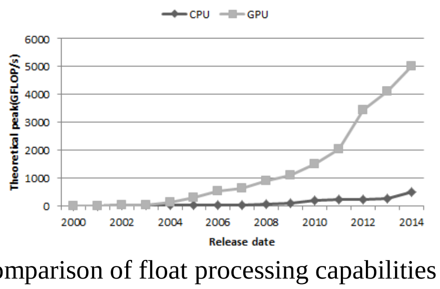

Fig. 1. Comparison of float processing capabilities between GPUs and CPUs

In contrast to the CPU, the GPU is more focused on data processing. It greatly reduces the logic control unit and cache, which typically requires significant hardware resources. This allows the processing capabilities on GPUs to exceed those of contemporary CPU products, as shown in Fig. 1. Therefore, GPUs play an increasingly important role in large data processing. Parallel accelerated applications, from simple graphics processing to signal processing, modeling, simulation, financial forecasting, and bio-computing, all require processing of large amounts of data. Niederhauser et al. [22] designed a parallel baseline drift removal method based on GPU that can efficiently process tens of hours of long-term ECG data.

NVIDIA introduced the Compute Unified Device Architecture (CUDA) in 2007, which allowed programmers to use C to easily develop parallel programs. In the CUDA program, the GPU is mapped to the structure shown in Fig. 2 and is abstracted into a grid. One grid is composed of several blocks, and each block contains many threads. These threads can be processed in parallel. Each thread can read data from memory that is used to perform a specific logic. The optimization of the number of threads in each block has become an important factor that affects the efficiency of program execution.

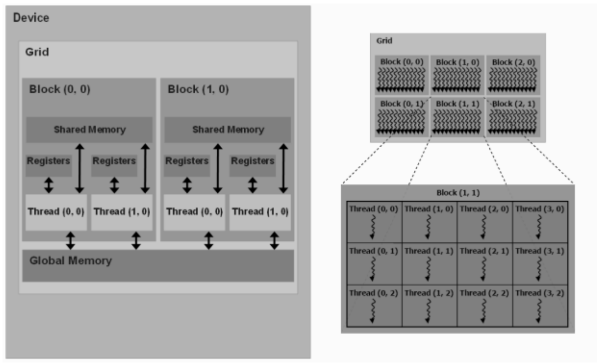

Fig. 2. CUDA programming model

When a CUDA program executes, many threads may need to access the same piece of memory space at the same time. Therefore, the access bandwidth has become another important factor that limits program efficiency. Therefore, when designing a CUDA program, rational optimization of memory access directly affects application performance and is the most difficult aspect of program optimization.

B. Feature Extraction

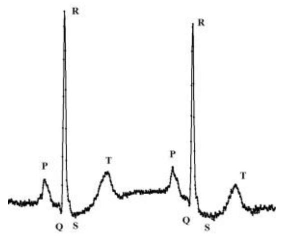

Fig. 3. The position of PQRST wave

To extract the features of ECG signal, we must first find the PQRST wave positions (Fig. 3) in the signal. The DOM (Difference Operation Method), which has been proposed in recent years, is an effective method to find the PQRST wave positions. More information about this algorithm is provided in [23]. Although the complexity of this method is O (N), this method is still time consuming when N is very large. To solve this problem and to improve the operation efficiency, we redesigned this method and made it operate in parallel with a large number of computing units in GPU. After found the PQRST wave positions, the eight heartbeat waveform features were calculated, as shown in Table I. The algorithm flow is shown in Fig. 4.

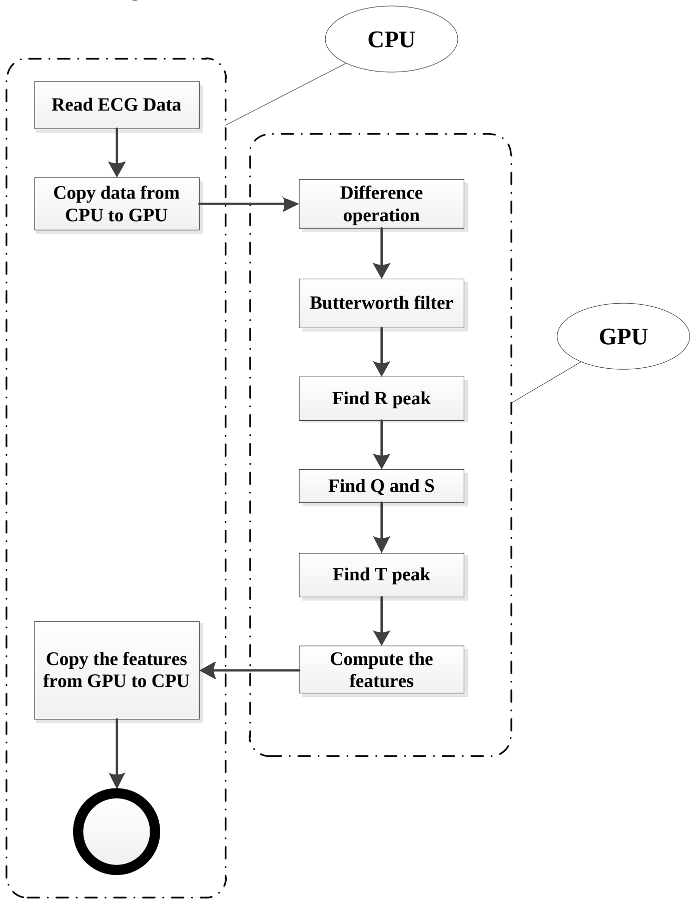

Fig. 4. Feature extraction process based on GPU

TABLE I
DESCRIPTION OF HEARTBEAT FEATURES

SortNo	Name	Description
1	HQR	The amplitude between Q and R in a QRS complex
2	HRS	The amplitude between R and S in a QRS complex
3	QRSdur	The time duration between Q and S in a QRS complex
4	RRdur	The time duration between an R peak and the next R peak
5	SlopeQR	The slope between Q and R in a QRS complex
6	SlopeRS	The slope between R and S in a QRS complex
7	SlopeST	The slope between S and T in a QRS complex
8	QTPint	The time duration between Q and T in a QRS complex

C. GRNN

Neural networks have many positive characteristics, such as nonlinearity, robustness, learning ability and massive parallel distributed processing. These traits allow neural networks to be widely applied in classification fields. Specht proposed the General Regression Neural Network (GRNN) in 1991 [24], which is a branch of the Radial Basis Function Neural Network where only the smoothing parameter requires optimization. GRNN has advantages, such as simple training and stable results. GRNN has been widely used for non-linear function approximation, data classification, forecasting, and among other applications. One study [25] used time-frequency analysis and GRNN to classify the crying of babies and further analyzed their mental and physical state with high classification accuracy. Another researcher [26] used GRNN to classify diabetes in Pima Indians and showed that GRNN possessed high accuracy and was an easily implementable method for solving medical data classification problems. The third group [27] used GRNN to forecast the relationship between quantitative structure and pharmacokinetics and achieved a good prediction accuracy.

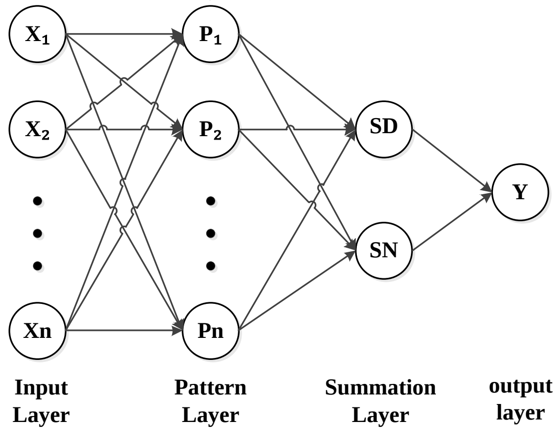

Fig. 5. The structure of GRNN

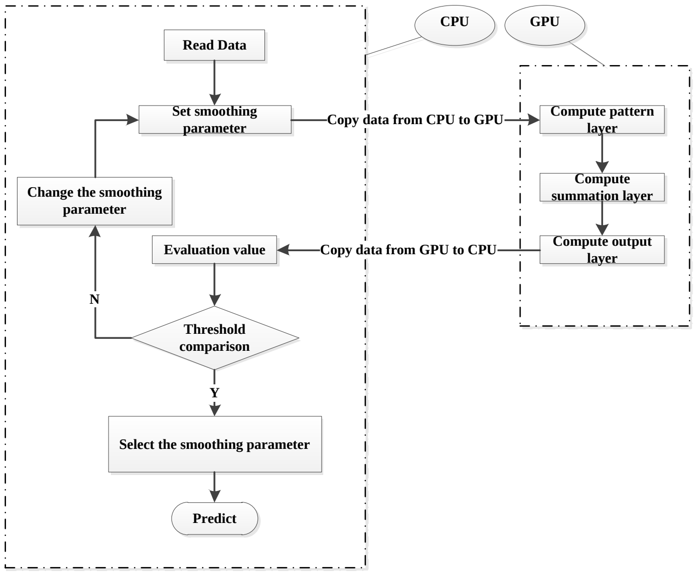

Fig. 6. Flowchart of the parallel GRNN based on a GPU

The structure of the GRNN is shown in Fig. 5. Before calculating the final GRNN output, we need to calculate the output of neurons in the pattern and summation layers. Let us define the number of datasets that we need to classify as N, the dimension of each sample as L, the computational complexity of pattern layer computation as O(NL) and the computational complexity of the summation layer computation as O(N). These steps are the algorithm core. However, they require the highest computational complexity in the GRNN and must be performed several times based on the objective value, causing the CPU-based GRNN algorithm to run slowly. To improve the efficiency of the algorithm, we propose to use a GRNN algorithm based on a GPU by using the multi-computing cores on the GPU to process in parallel the samples in a dataset. The algorithm flowchart is shown in Fig. 6.

First, the sample is read into the CPU. We define the number of samples as N and the dimension of each sample as L. We constitute the sample space as $X = \{X_1, X_2, X_3, \cdots, X_N\}$ in which X_i is the i-th sample in the dataset (1), and we select 1-10 as ten numbers in the first group's smoothing parameters.

$$X = \begin{Bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & \cdots & x_{1,L} \\ x_{2,1} & x_{2,2} & x_{2,3} & \cdots & x_{2,L} \\ x_{3,1} & x_{3,2} & x_{3,3} & \cdots & x_{3,L} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & x_{N,3} & \cdots & x_{N,L} \end{Bmatrix} \quad (1)$$

Then, memory space is allocated in the GPU device. The GPU will process multiple adjacent stored data when threads access the global memory. Therefore, the GPU only requires access to the global memory when threads read the adjacent data. Because of this feature, the CPU matrix is copied to global memory on the GPU after transposition, which accelerates the speed of memory access. A GPU matrix is generated as X_{GPU} to store the sample data. Then, the GPU begins to perform the computing tasks, and the CPU is only used to control the changes in the smoothing parameters and number of iterations.

One neuron in the pattern layer corresponds to a learning sample. The transfer function of the i-th neuron is shown in (2) in which X_{in} is the input of the network, X_i is the learning sample corresponding to the i-th neuron, and σ is the smoothing parameter.

$$p_i = \exp\left[-\frac{(X_{in}-X_i)^T(X_{in}-X_i)}{2\sigma^2}\right] \quad (2)$$

The summation layer includes two types of neurons. One is used to accumulate all of the output of the neurons in the pattern layer, and its transfer function is shown in (3). The other neuron type is used to produce a weighted summation of all of the neuronal output in the pattern layer, and its transfer function is shown in (4). The connection weight between the i-th neuron in the pattern layer and the j-th neuron in the summation layer is the j-th element in the output of the i-th learning sample.

$$S_D = \sum_{i=1}^{n} p_i \quad (3)$$

$$S_{Nj} = \sum_{i=1}^{n} y_{ij} p_i \quad (4)$$

The number of neurons in the output layer is the output vector dimension of the learning sample. The j-th neuron corresponds to the j-th element of the output and is calculated as shown in (5).

$$y_j = \frac{S_{Nj}}{S_D} \quad (5)$$

This study designed one block that is responsible for one sample classification such that each block must complete the calculations in Equations 7-10. Each block was designed to include TPB threads, where TPB=Kernels $\times$ p,($1 \leqslant p \leqslant$ MaxThread/48) threads in one block. The MaxThread is the maximum number of threads per block. The kernels are determined by the GPU hardware architecture. A kernel is the number of operational units included in one Streaming Multiprocessor (SM) in the GPU and is also the number of parallel threads when the program is executing. Designing the TPB to be an integer multiple of kernels fully exploits the parallel computing capabilities of each SM. All of the threads in the same block used the same input sample. L threads read the input sample from the global memory to the shared memory in each block, which can accelerate the data access speed. Each thread must complete the N/TPB output computations of the pattern layer, sum them, and then use the reduction summation method to accumulate the results of all of the threads and to obtain the outputs of the summation layer. Finally, each thread calculate the classification results of the input sample according to Equation 10. Because the size of the sample space is N, N blocks are required to complete the classification of the entire sample space.

After the classification of the sample space, the results were copied from the GPU back to the CPU, and the classification accuracy is calculated in the CPU. When training this model, we used ten smoothing parameters to classify the sample space and record the classification accuracy. We selected the smoothing parameter with the highest classification accuracy as the best smoothing parameter. The data accuracy of the best smoothing parameter and the preset threshold were compared. If the threshold was reached, then the training was completed. If not, then one digit was added to the best smoothing parameter, and ten floats were selected using the smoothing parameter as the center. For example, if the best smoothing parameter was 2.7, then the next ten floats selected would be 2.66, 2.67, 2.68, 2.69, 2.70, 2.71, 2.72, 2.73, 2.74, and 2.75. These ten floats would be used as the new group of smoothing parameters to classify the sample space again, and the training would proceed.

D. *Personalized Classification Model*

The classification accuracy of the GRNN model significantly depends on the coverage of the samples in the pattern layer. If the sample coverage is larger, the accuracy that the model can achieve will be greater. Thus, the achievement of a personalized heartbeat classification model is equal to the construction of a personalized sample library in the pattern layer during the model training process. When a test sample was classified in the wrong type (named as ERR_SAMPLE), we put this ERR_SAMPLE into the pattern layer. Then we need to remove an old sample in the pattern layer witch had the same label as the ERR_SAMPLE and had the longest Euclidean distance to the ERR_SAMPLE. Thus, the samples in the pattern layer were more and more similar to the test samples, and a personalized samples library could be constructed. In practical applications, we usually must update many samples at the same time. At this time, we must process this problem parallel to complete the update of the model in a short time.

III. Results

A. *Experimental Environment*

The experimental environment was an ordinary PC computer with a CPU Intel Core i3-3240. The frequency of the CPU was 3.4 GHz. The GPUs included a NVIDIA GeForce GTX620, a NVIDIA GeForce GTX660 and a NVIDIA GeForce GTX780Ti. Table II shows the parameters of the three GPUs. The programming language was CUDA_C, and the debugging environment was Microsoft Visual Studio 2010.

Table II
THE GPU PARAMETERS

Parameters	GTX620	GTX660	GTX780Ti
Memory (MBytes)	1024	2048	3072
Memory bandwidth	64-bit	192-bit	384-bit
Memory clock rate	1.62 GHz	3.004 GHz	3.5 GHz
CUDA Cores	48	960	2880
Maximum number of threads per block	1024	1024	1024
Total amount of shared memory per block (Bytes)	49152	49152	49152
Registers available per block	32768	65536	65536
CUDA version 5.5	5.5	6.0	6.0

B. Base Classification

Table III
CONFUSION MATRICES OF THE BASE ECG BEAT CLASSFICATION RESULTS

	N	S	V	F	Q
N	9147	94	167	57	2
S	87	807	16	1	0
V	96	2	1511	28	3
F	73	3	20	439	0
Q	0	3	0	0	16
Acc	0.97	0.89	0.89	0.84	0.77

In this experiment, we selected seventeen 30 min-long ECG signal groups from the MIT-BIH arrhythmia database [28] as the training data set. The serial numbers of these records were 100, 103, 104, 106, 112, 119, 122, 200, 203, 208, 209, 217, 222, 223, 230, 232, and 233. The data sampling rate was 360 Hz. According to the AAMI recommendation, the experimental task was to classify the 5 types of heartbeat: N, S, V, F and Q. According to the labels on the ECG data, we found a total of 18805 N, 1817 S, 3427 V, 1049 F, and 41 Q. We segmented the heartbeats and extracted 8 features from each heartbeat; we then randomly selected 50% of the data as the training set and the other 50% as the test data. The performance of the GRNN is shown in Table III. The average accuracy achieved 95%.

Then, we measured the ability of the classification algorithm to distinguish certain events (i.e., VEBs or SVEBs) from the nonevents (i.e., non-VEBs or non-SVEBs). The classification performance was measured using the four standard metrics found in the literature [29]: classification accuracy (Acc), sensitivity (Sen), specificity (Spe), and positive predictivity (Ppr). The respective definitions of these four common metrics using true positive (TP), true negative (TN), false positive (FP), and false negative (FN) are as follows. The accuracy is the ratio of the number of correctly classified patterns to the total number of patterns classified, Acc = (TP+TN)/(TP+TN+FP+FN). The sensitivity is the rate of correctly classified events among all of the events, Sen = TP/(TP+FN). The specificity is the rate of correctly classified nonevents among all of the nonevents, Spe = TN/(TN+FP). Finally, the positive predictivity is the rate of correctly classified events in all of the detected events, Ppr = TP/(TP+FP). These are shown in Table IV.

Table IV
THE CLASSIFACATION PERFORMANCE IN VEB AND SVEB

VEB				SVEB			
TP	1509	Acc	0.989155	TP	775	Acc	0.993725
TN	10713	Sen	0.880397	TN	11577	Sen	0.855408
FP	121	Spe	0.988831	FP	65	Spe	0.994417
FN	205	Ppr	0.925767	FN	131	Ppr	0.922619

C. Personalized Classification

During the process of online learning, we used 300 real patients' holter data from the Navy General Hospital in Beijing. The data sampling rate was 200 Hz. Each patient's record was approximately 24 hours long and included approximately 120,000 heartbeats. Because the signals were too long, the doctors did not mark all the heartbeats. They only marked a few abnormal heartbeats and made a diagnosis according to these marked heartbeats. In these 300 patients, there were 136 patients' ECG that contained SVEB heartbeats and another 164 containing VEB heartbeats. As long as the classification model for determining a patient's data primarily contained SVEB heartbeats, the patient was classified as SVEB class. This was the same for VEB.

During the classification process, because the data sampling rate from the Navy General Hospital was not the same as the MIT-BIH data, we changed the sample frequency to 360 Hz via an interpolation calculation. Then, we extracted the features. Directly using the classifier obtained using the above training, the classification accuracy rate was only 76%. According to the heartbeats marked by the doctors, we used the method introduced in section II to update the model. We received an automatic personalized classification model for each patient. After the update, the accuracy increased to 88%. The results are shown in Table V.

Table V
THE CLASSIFICATION RESULTS OF THE MODEL AFTER THE UPDATE

	SVEB	VEB
N	20	13
SVEB	114	4
VEB	2	147

D. Program Efficiency

To simplify the process of debugging, we compared the program parallelization parameters, such as the number of parallel threads, the use of the shared memory, the design of data type, etc., under GTX620. After the parameters were determined, we ran the program under GTX620, GTX660 and GTX780Ti and compared the efficiencies. With the increase in the data required for processing, the advantages of GPU parallel processing became more obvious.

Regarding feature extraction, the DOM was used to extract the features of the ECG signal and to redesign in parallel the method under the GPU. Fig. 7 shows the time consumed by the extraction method under the CPU and GPU, and the horizontal coordinates are the number of data groups used in the experiment (from 1 to 9). GPU improved the efficiency by approximately 28-fold when processing 9 groups of data simultaneously.

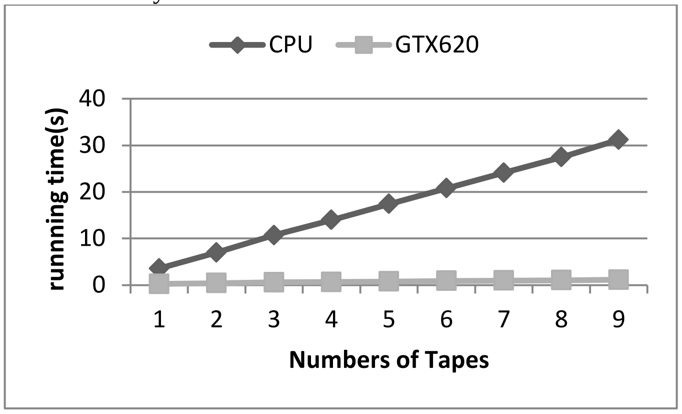

Fig. 7. Comparison of the feature extraction

Regarding model training, the impact of the program efficiency due to the number of threads in each block and the use of shared memory are important to discuss. According to the previously described formula, $TPB = kernels \times p$, which can be used to select the number of threads in each block. The GPU used in this experiment was a GTX620 and only had one SM that included 48 CUDA cores. Therefore, $kernels = 48$, and the parameter, p, was selected. This study selected $p = 1, 2, 4, 8$ to experimentally evaluate the differences between these values. A comparison of the execution time of the parallel GRNN with the use of shared memory or not in the 103 datasets is shown in Fig. 8.

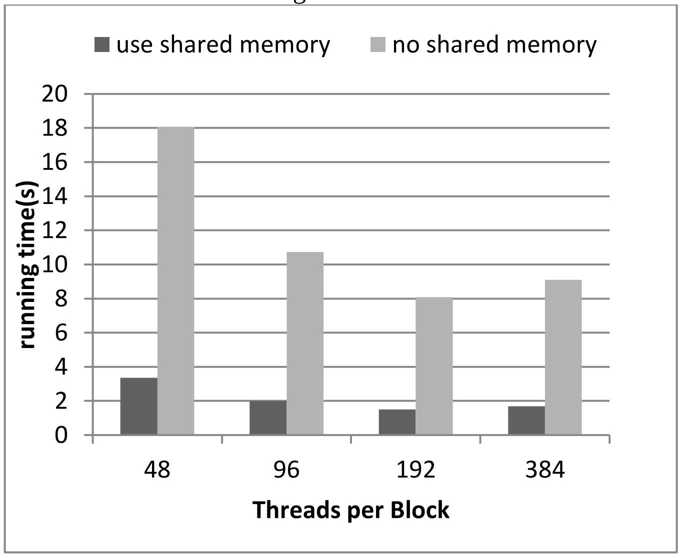

Fig. 8. The impact of the program efficiency, which based on the number of threads in each block, as a function of the use of shared memory

To determine which data type was used in the program, we compared the accuracy and the efficiency of the classification model under float and double, respectively. The test results show that it was slower when a double variable was used in the program, and the classification accuracy presented no obvious change. Thus, in the next experiment we only used float variables in the program.

In this experiment, we used the 8 features extracted previously to classify the heartbeat dataset into different sizes. We tested the efficiency of the parallel GRNN that was based on the GPU and the serial GRNN that was based on the CPU for these ten datasets, and the results are shown in Fig. 9. The GPU-accelerated effect was higher at lower sample dimensions by approximately 23-fold. The growth of the dimensions reduced the GPU-accelerated effect because it was limited by the hardware. However, using the GPU was still more than 20-fold faster than the serial algorithm that was based on the CPU.

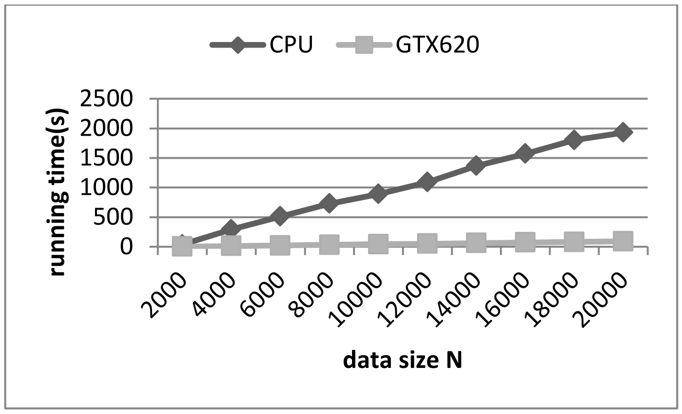

Fig. 9. Comparison of the GRNN efficiency between the GPU and the CPU

Here, we compared the CPU and three different GPUs for different data sizes, as shown in Fig. 10. We achieved an approximately 450-fold increase in performance speed using the GTX780.

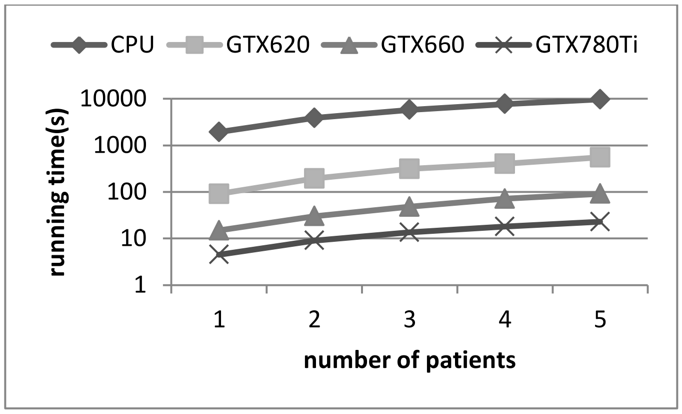

Fig. 10. The efficiency of the GRNN for different hardware

IV. Discussion

We designed an effective heartbeat classification method based on a GPU, which included a parallel DOM to extract the features of heartbeats and a parallel GRNN to classify the heartbeat. We added an online-learning module into the parallel GRNN and made it able to update the samples in the pattern layer according to the actual situation of the patients.

A. *Base Classification*

We constructed a personalized automatic classification model for each patient and obtained high classification accuracy. The parallel GRNN in this paper achieved a 95% accuracy in the MIT-BIH arrhythmia database. Compared with the models in [12-21], the accuracy of our model was slightly lower. The comparison is shown in Table VI. For the classification of a certain kind of abnormal heartbeat, Table VII shows the results of this experiment and the comparison with other studies [17, 29-31]. The Sen and Ppr in VEB were lower than the best research result to date. However, the Sen in SVEB was significantly higher compared with the other research.

Table VI
COMPARISION WITH THE RECENT RESULTS

Author	Algorithm	Accuracy
L. Mariano [12]	linear discriminant functions	93%
Julien Oster [13]	novel detection based on switching kalman filters	98%
L. de Oliveira [14]	dynamic bayesian network	95%
G. de Lannoy [15]	weighted conditional random fields	85%
X. Jiang [16]	independent component analysis	98%
T. Ince [17]	particle swarm optimization	98%
M. Lagerholm [18]	self-organizing maps	98%
V. S. R. Kumari [19]	multilayer perceptron neural network	95%
C. V. Banupriya [20]	probabilistic neural network	98%
L. S. de Oliveira [21]	dynamic bayesian network	95%
Proposed	general regression neural network	95%

TableVII
VEB AND SVEB CLASSIFICATION PERFORMANCE OF THE PROPOSED METHOD AND ITS COMPARISION WITH THE FOUR MAJOR ALGORITHMS FROM THE LITERATURE

Author	VEB				SVEB			
	Acc	Sen	Spe	Ppr	Acc	Sen	Spe	Ppr
Hu et[29]	94.8	78.9	96.8	75.8	N/A	N/A	N/A	N/A
Jiang[30]	98.8	94.3	99.4	95.8	97.5	74.9	98.8	78.8
Ince[17]	97.9	90.3	98.8	92.2	96.1	81.8	98.5	63.4
Kiranyaz[31]	98.9	95.9	99.4	96.2	96.4	68.8	99.5	79.2
Proposed	98.9	88.0	98.9	92.6	99.4	85.5	99.4	92.3

Compared to the other neural networks in [18-21], although the accuracy of our model was slightly lower, GRNN has certain advantages in some other aspects. If we need to update the SOMNN model in [18] or the MLPNN model in [19], all the samples in new training set must be involved in the calculation of retraining. In GRNN, we only need to put the new samples into the pattern layer and remove some old samples, a retraining is not necessary. The PNN used in [20] is similar to GRNN in principle. But GRNN only has linear neurons, PNN has some competitive neurons. So GRNN can be parallel processed easily. The update training of the dynamic bayesian network in [21] is also more complex than GRNN. So we choose GRNN as the personalized classification model.

B. Personalized classification

In the practical application of the model, our GRNN classification model will keep updating according to the specific ECG data of the patient. Based on a fixed model, our model become a customized model for the patient. Using practical verification, the average accuracy of VEB and SVEB achieved was 88%. Compared with the fixed model, the updated model improved the accuracy by 12%. Most research on heartbeat classification is not applied to the actual data. Only [17, 29-31] proposed an adaptive model, which is based on the individual data of patients. The accuracy was above 90% in the abovementioned papers. However, the data in their tests were derived from MIT-BIH arrhythmia data. The data in our study originated from a hospital without any preprocessing. Thus, mistakes can appear during the process of heartbeat segmentation, feature extraction and classification. This is the reason that may cause poor performance of the classification model in a practical application.

C. Program Efficiency

The classification accuracy of the GRNN model largely depends on the coverage of samples in the pattern layer. The higher the sample coverage, the higher accuracy of the model can be obtained. However, the larger the size of the pattern layer, the heavier the computational burden of the program. Specifically, a one day holter data of a patient usually contains hundreds of thousands of heartbeats. This large amount of data results in a high computational pressure on the classification model. To solve this problem, a parallel GRNN model was redesigned under the CUDA framework. Using many computing cores of a GPU to process the data in parallel, the efficiency was significantly improved. Compared with the results in [32], the efficiency of the proposed methods is significantly higher, and the acceleration effect is more stable between different dataset sizes. Compared with the CPU-based methods that are mentioned in [12-21], our method is based on a GPU and is clearly more efficient.

Regarding the feature extraction, we run the program on the GT620 with 48 CUDA cores, which resulted in a 28-fold efficiency improvement. Approximately 1 s was needed to process 9 data groups at the same time. Regarding the model training, we used a shared GPU memory and thread number in each block to perform the experiment. It is clear from Fig. 8 that when the thread number in each block was 192, and the shared memory was used, the efficiency was the highest. We can complete the training or update the calculation in 2 s using GT620. However, if only CPU was used, it would have required more than 10 times that time.

It is worth mentioning that in terms of data classification, we save more time using a parallel program. Using the serial design procedure, it will require more than 30 min to classify the ECG data of one patient in one day (more than 100000 heartbeats). Under the same conditions, it will require 90 s for GT620, 15 s for GTX660, and 5 s for GTX780Ti. This program is much more efficient than the program in [31] under the OpenMP framework. If there are 10 patients with dynamic ECG monitoring, it takes more than 5 hours a day to analyze the data using a serial design. In our parallel design, it takes less than 1 minutes to complete all computing tasks. We can use a common PC with a GPU to improve the efficiency of ECG data processing by several hundred times, which will significantly improve the work efficiency of doctors.

After the comparison, the results are encouraging. The proposed methods are competitive and have clear advantages in speed and classification accuracy. When dealing with the large amount of data problem, such as long–term ECG data, the strong computing power our methods displayed can save significant amount of time for the doctors.

V. Conclusion

This paper implemented an effective GPU-based heartbeat classification method that included a parallel DOM to extract heartbeat features and a parallel GRNN to classify the heartbeats. According to AAMI, the achieved accuracy was 95%. We added an online-learning module into the parallel GRNN and constructed a personalized automatic classification model for each patient. The classification accuracy was 88% for the real patient holter data. This paper redesigned the original algorithm and made the program execute in parallel on numerous computational cores of a GPU. Compared with a serial algorithm that is based on a CPU, the parallel GPU-based algorithm improved the efficiency by approximately hundred times. This significantly shortens the data processing time and saves a significant amount of time for the doctors. With further development of medical electronics, the recording time of a holter will increase, and the sampling rate will become higher and higher, which will result in a large amount of accumulated data in databases. Data analysis is becoming increasingly difficult, and the time cost is also significantly increasing. The experiments in this paper show that the parallel programming model, which is based on a GPU, provides an efficient and generic way to analyze large datasets. The algorithm presented in this paper achieved an approximately 450-fold increase in performance speed using a GTX780Ti. When a high-level GPU is used, the algorithm has the potential to achieve even better performance speeds with a slight adjustment of the TPB parameter. Therefore, the parallel algorithm, which is based on a GPU, has broad applications. Doctors can use a normal PC to complete high performance computing without applying a large-scale computing server. This improves the efficiency of work for the doctors and reduces the computing cost in the hospital.

References

[1] J. A. Kastor, London, U.K.: W.B. Saunders,1994.
[2] P. de Chazal, "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," Ieee Transactions on Biomedical Engineering, vol. 51, no. 7, pp. 1196-1206, Jul, 2004.
[3] M. Milanesi, "Independent component analysis applied to the removal of motion artifacts from electrocardiographic signals," Med. Biol. Eng. Comput., vol. 46, no. 3, pp. 251–261, Mar. 2008.
[4] O. Sayadi and M. B. Shamsollahi, "Model-based fiducial points extraction for baseline wandered electrocardiograms," IEEE Trans. Biomed. Eng.,vol. 55, no. 1, pp. 347 –351, Jan. 2008.
[5] S. Ari, "ECG signal enhancement using S-Transform," Comput. Biol. Med., vol. 43, no. 6, pp. 649–660, Jul. 2013.
[6] M. Zivanovic and M. Gonz ́alez-Izal, "Simultaneous powerline interference and baseline wander removal from ECG and EMG signals by sinusoidal modeling," Med. Eng. Phys., vol. 35, no. 10, pp. 1431–1441, Oct. 2013.
[7] A. Fasano and V. Villani, "Baseline wander removal for bioelectrical signals by quadratic variation reduction," Signal Process., vol. 99, pp. 48–57, Jun. 2014.
[8] M. Merino, "Envelopment filter and K-means for the detection of QRS waveforms in electrocardiogram," Medical Engineering & Physics, vol. 37, no. 6, pp. 605-609, Jun, 2015.
[9] C. Ye, "Heartbeat Classification Using Morphological and Dynamic Features of ECG Signals," Biomedical Engineering, IEEE Transactions on, vol. 59, no. 10, pp. 2930-2941, 2012.
[10] C. A. Bustamante, "ECG delineation and ischemic ST-segment detection based in wavelet transform and support vector machines." pp. 1-7.
[11] Y.-C. Yeh, "Feature selection algorithm for ECG signals using Range-Overlaps Method," Expert Systems with Applications, vol. 37, no. 4, pp. 3499–3512, 2010.
[12] L. Mariano and M. Juan Pablo, "Heartbeat classification using feature selection driven by database generalization criteria," IEEE Trans Biomed Eng, vol. 58, no. 3, pp. 616 - 625, 2011.
[13] Oster J, Behar J, Sayadi O, et al. Semi-supervised ECG Ventricular Beat Classification with Novelty Detection Based on Switching Kalman Filters.[J]. IEEE transactions on bio-medical engineering, 62(9):2125-2134, 2015.
[14] L. de Oliveira, "Premature Ventricular beat classification using a dynamic Bayesian network," in Proc. IEEE Int. Conf. Eng. Med. Biol. Soc., Aug./Sep. 2011, pp. 4984–4987.
[15] G. de Lannoy, "Weighted conditional random fields for supervised interpatient heartbeat classification," IEEE Trans. Biomed. Eng., vol. 59, no. 1, pp. 241–247, Jan. 2012.
[16] X. Jiang, "ECG arrhythmias recognition system based on independent component analysis feature extraction," in Proc. IEEE Region 10 Conf., pp. 1–4, Nov. 2006.
[17] T. Ince, "A generic and robust system for automated patient-specific classification of ECG signals," IEEE Trans. Biomed. Eng., vol. 56, no. 5, pp. 1415–1426, May 2009.
[18] M. Lagerholm, "Clustering ECG complexes using Hermite functions and self-organizing maps," IEEE Trans. Biomed. Eng., vol. 47, no. 7, pp. 838–848, Jul. 2000.
[19] V. S. R. Kumari and P. R. Kumar, "Cardiac Arrhythmia Prediction using improved Multilayer Perceptron Neural Network," Research and Develop. (IJECIERD), vol. 3, no. 4, pp. 73-80, 2013
[20] C. V. Banupriya and S. Karpagavalli, "Electrocardiogram Beat Classification using Probabilistic Neural Network, " Int. J. of Comput. Applicat. (IJCA), pp. 31-37, 2014.
[21] L. S. de Oliveira, "Premature Ventricular beat classification using a dynamic Bayesian Network." pp. 4984 - 4987.
[22] T. Niederhauser, "Graphics-Processor-Unit-Based Parallelization of Optimized Baseline Wander Filtering Algorithms for Long-Term Electrocardiography," Ieee Transactions on Biomedical Engineering, vol. 62, no. 6, pp. 1576-1584, Jun, 2015.
[23] Y.-C. Yeh and W.-J. Wang, "QRS complexes detection for ECG signal: the Difference Operation Method," Computer Methods\s&\sprograms in Biomedicine, vol. 91, no. 3, pp. 245-54, 2008.
[24] D. F. Specht, "A general regression neural network," Neural Networks, IEEE Transactions on, vol. 2, no. 6, pp. 568-576, 1991.
[25] M. Hariharan, "Normal and hypoacoustic infant cry signal classification using time–frequency analysis and general regression neural network," Computer methods and programs in biomedicine, vol. 108, no. 2, pp. 559-569, 2012.
[26] K. Kayaer and T. Yıldırım, "Medical diagnosis on Pima Indian diabetes using general regression neural networks." pp. 181-184,2003.
[27] C. Yap and Y. Chen, "Quantitative structure–pharmacokinetic relationships for drug distribution properties by using general regression neural network," Journal of pharmaceutical sciences, vol. 94, no. 1, pp. 153-168, 2005.
[28] R. Mark and G. Moody, "MIT-BIH Arrhythmia Database Directory," [Online]. Available: http://ecg.mit.edu/dbinfo.html
[29] Y. Hu, "A patient-adaptable ECG beatclassifier using a mixture of experts approach," IEEE Trans. Biomed. Eng., vol. 44, no. 9, pp. 891–900, Sep. 1997.
[30] W. Jiang and S. G. Kong, "Block-based neural networks for personalized ECG signal classification," IEEE Trans. Neural Networks,vol. 18, no. 6, pp. 1750–1761, Nov. 2007.
[31] S. Kiranyaz, "Real-time patient-specific ECG classification by 1D convolutional neural networks," 2015.
[32] R. Czabański, "Fetal state evaluation with fuzzy analysis of newborn attributes using CUDA architecture," Journal of Medical Informatics & Technologies, vol. 22, 2013.