A Novel Fast Training Method for SVM and its Application in Fault Diagnosis of Service Robot

http://dx.doi.org/10.3991/ijoe.v11i6.4846

Xianfeng Yuan, Mumin Song, Fengyu Zhou*, Yugang Wang, and Zhumin Chen
Shandong University, Jinan, China

***Abstract*—Support Vector Machines (SVM) are popular machine learning algorithms that have been successfully applied in diverse aspects, but for large training data sets the processing time and computational costs are prohibitive. This paper presents a novel fast training method for SVM, which has been applied in the fault diagnosis of a service robot. First, sensor data were sampled under different running conditions of the robot and those samples were divided as training sets and testing sets. Second, the sampled data were preprocessed and a principal component analysis (PCA) model was established for fault feature extraction. Third, the feature vectors were used to train the SVM classifier, which achieved the fault diagnosis of the robot. To speed up the training process of the SVM, on the one hand, sample reduction was done using the proposed support vectors selection (SVS) algorithm, which can ensure good classification accuracy and generalization capability. On the other hand, we took advantage of the excellent parallel computing abilities of a Graphics Processing Unit (GPU) to pre-calculate the kernel matrix, which avoids recalculation during the cross validation process. Experimental results illustrate that the proposed method can significantly reduce the training time without decreasing the classification accuracy.**

***Index Terms*—SVM, fast training method, support vectors selection, GPU, robot fault diagnosis.**

I. INTRODUCTION

Robots are playing more and more important roles in our daily life [1], particularly in the home service area [2-3]. However, most of the robot users are ordinary people or even the frail elderly who have low self defense abilities, which implies that the faults occured in the robot system are serious threats to the user's property and life. Hence, it is meaningful to do study on the fault diagnosis methods (FDMs) for service robots that work in a man-robot coexistent environment.

In general, existing FDMs for robots can be divided into two major types: model based and data driven. In [4], a model based discrete-time fault diagnosis framework for the manipulator is presented. Yu et al. [5] introduced a robot fault-proneness prediction method based on particle filters. Moreover, Hoang et al. [6] developed a nonliner observer-based fault diagnosis framework for wheeled mobile robots. Those model-based methods are effective, but it is not easy to get an accurate mathematical model of a service robot that works in a dynamic environment. On the contrary, data-driven FDMs can achieve good diagnosis performance without an accurate mathematical model. Machine learning based fault diagnosis methods are usually regarded as the most typical data-driven FDMs [7]. As one of the state-of-the-art machine learning algorithms, SVM has shown potential and promising performance in classification and has been successfully applied in many fault diagnosis occasions [8-9], which include robot fault diagnosis. However, the standard SVM algorithm consists of solving a mathematical optimization problem of Quadratic Programming (QP), and its computational cost is at least $O(n^2)$, where n is the number of training vectors. Thus, the training speed of SVM for large datasets becomes a bottleneck.

To accelerate the training speed of an SVM, intensive research efforts have been made [10]. An intuitive solution to reduce the training time is to decompose the whole optimization problem into several sub-problems, and thus the overall training time can be reduced [11]. However, the hyper plane constructed by SVM is only dependent on a small number of training samples named as support vectors, which lie close to the decision boundary. Inspired by this idea, clustering algorithms [12] and random sampling algorithms [13] can be applied in SVM to select the support vectors from the whole dataset. Moreover, GPU-based parallel SVM training methods [14-15], which are much faster than the traditional CPU-based serial ones, have been proposed. Although the above mentioned methods are effective to some extent, there are still several problems in those methods: (1) the sample reduction based SVM training methods are usually designed for binary classification, but for multi-classification tasks, the computational costs of the sample reduction process are huge and (2) the loss of support vectors caused by the sample reduction process is negative for the generalization capability of SVM.

To cope with the above mentioned problems, this paper presents a novel fast training method for SVM. Considering that the training vectors that lie near the decision hyperplane are likely to become support vectors, we proposed a novel support vectors selection (SVS) algorithm to preserve the support vectors and delete the redundant training vectors that are far from the margins. First, the C-means clustering algorithm was applied to detect the C cluster centers. Second, the clusters with more than one class label (named as CML) are preserved, while the others (named as CSL) are selected to be further analyzed. Third, the k nearest clusters of each CSL are preserved and applied as the training vectors together with the CML. With the proposed SVS algorithm, we can accomplish the sample reduction procedure with the lowest loss of the support vectors, which implies that the proposed method can guarantee the accuracy and generalization of SVM. Further, we pre-calculated the kernel matrix to avoid recalculation during the cross validation. Taking advantage of the parallel computing techniques, the pre-calculation

was implemented on a GPU, which contributed to the reduction of the SVM training time. To verify the effectiveness of the proposed method, an application in robot fault diagnosis was demonstrated and experimental results are given.

The remainder of this paper is organized as follows. Section II briefly introduces the SVM theory. Section III presents the proposed fast training algorithm for SVM. Experimental results are given in Section IV. Finally, Section V is devoted to conclusions.

II. SUPPORT VECTOR MACHINES

The basic SVM deals with linearly separable binary classification problems and can cope with the non-linearly separable cases by introducing the kernel fuctions and slack penalty. Given a labeled training set consisting of a group of data points $S_t = \{x_i, y_i\}_{i=1}^{n}$, where $x_i \in R^m$ is the ith training input vector, n is the number of training data, m is the dimension of the input data, and $y_i \in \{-1, 1\}$ is the set of labels. The SVM training problem can be written as the following Quadratic Program:

$$\begin{cases} \min\limits_{\alpha_i} \left(\sum\limits_{i=1}^{n} \alpha_i - \frac{1}{2} \sum\limits_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j x_i^T x_j \right) \\ \text{s.t.} \sum\limits_{i=1}^{n} \alpha_i y_i = 0, \quad 0 \le \alpha_i \le c, \quad i = 1, \mathrm{K}\ , n \end{cases} \tag{1}$$

where α_i is the Lagrangian coefficient and c is the slack penalty.

The kernel function can map the input vector x into feature spaces and returns a dot product of the feature space. The liner discriminant function with kernel $K(x_i, x_j)$ is given by the following:

$$f(x) = \mathrm{sgn}\left(\sum_{i,j=1}^{n} \alpha_i y_i K(x_i, x_j) + b \right) \tag{2}$$

where $\mathrm{sgn}(x)$ is the signum function and the structure of SVM classifier is shown in Fig. 1.

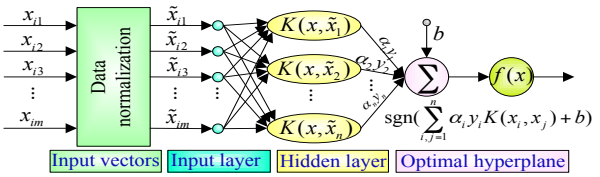

Figure 1. Structure of SVM

III. Fast Training Method for SVM

The proposed fast training method for SVM mainly contains two processes, namely, the support vectors selection (SVS) process and the GPU-based acceleration process. In the former process, the *C*-means clustering algorithm is applied to find the *C* cluster centers, and those clusters are divided as the CML and the CSL. Second, the CML are preserved as they are more likely to become support vectors, while the CSL are selected to be further analyzed. Third, the k nearest clusters of each CSL are preserved and applied as the traning samples together with the CML. Thus, sample reduction can be accomplished using the proposed SVS algorithm, which retains the support vectors to the utmost extent. In the latter process, a GPU-based pre-calculation of the kernel matrix is done to avoid the recalculation during the cross validation.

A. Support Vectors Selection Algorithm

Step1. Based on the *C*-means clustering algorithm, the original training samples are divided into *c* clusters, from which we can easily distinguish the CML and the CSL. The details are described in Table 1.

TABLE I.
PSEUDO CODE OF PROCEDURES IN FINDING CML AND CSL

Finding the CML and the CSL (x_i , c, N)

1. Randomly select a set of initial candidates $\{v_1, v_2, \mathrm{K}\ , v_c\}$ for c cluster centers from the training samples $X = \{x_1, x_2, \mathrm{L}\ , x_n\}$.
2. for p=1 to N do
3. Calculate the Euclidean distance between each $x_i (i = 1, 2, \mathrm{K}\ , n)$ and $v_j (j = 1, 2, \mathrm{K}\ , c)$, then assign each x_i to the nearest cluster C_j whose center is v_j.
4. Recompute the centers of the c clusters $C_j (j = 1, 2, \mathrm{K}\ , c)$ and we can get the new cluster centers:
$$v_j' = \frac{1}{n_j} \sum_{x \in C_j} x \quad (j = 1, 2, \mathrm{K}\ , c) \tag{3}$$
where n_j is the number of samples in cluster C_j and the sum of square error is calculated by (4).
$$E = \sum_{j=1}^{c} \sum_{x_i \in C_j} \mathrm{P} x_i - v_j \mathrm{P}^2 \quad (i = 1, \mathrm{K}\ , n;\ j = 1, \mathrm{K}\ , c) \tag{4}$$
5. if no more reassignment of samples or E is below a threshold
6. break.
7. end for
8. c clusters $C = \{(x_1, y_1), (x_2, y_2) \mathrm{L}\ , (x_c, y_c)\}$ can be got, where (x_i, y_i) represents the ith cluster center x_i with label y_i and $C = C_{csl} \mathrm{U}\, C_{cml}$.
9. Return C_{csl} and C_{cml}.

Step2. To retain support vectors as much as possible, we needed to do analysis on the CSL, which can be denoted by $C_{csl} = \{(x_1, y_1), (x_2, y_2) \mathrm{L}\ , (x_m, y_m)\}$. Then we found the k nearest cluster sets of x_i (marked as $C_k(x_i)$) by solving the following problem:

$$\begin{cases} \min \sum\limits_{j=1}^{k} \mathrm{P} x_i - x_j \mathrm{P}^2 \\ \text{s.t.} \quad y_j \ne y_i;\ x_i, x_j \in C_{csl} \end{cases} \tag{5}$$

Step3. The final training datasets can be found through the following equation

$$C_T = C_{cml} \mathrm{U}\, C_k(x_1) \mathrm{U}\, C_k(x_2) \mathrm{UL}\ \mathrm{U}\, C_k(x_m) \tag{6}$$

where C_T represents the training sets after sample reduction, C_{cml} represents the set of CML, and $C_k(x_i)$ repre-

sents the k nearest cluster sets of x_i .GPU-based Acceleration Process

The calculation of the kernel matrix (KM) is one of the most time consuming procedures in the training process of SVM. The kernel matrix will be recalculated k times during the k-fold cross validation, which increases the training time enormously. To avoid the recalculation and accelerate the calculation process of KM, a GPU-based precalculation of KM was proposed [16], which is shown in Fig. 2.

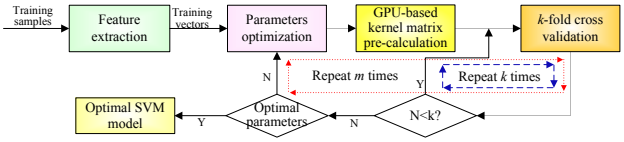

Figure 2. Proposed training procedure based on GPU

In this study, the Gaussian RBF kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ was calculated by

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma \, \mathrm{P}\boldsymbol{x}_i - \boldsymbol{x}_j \, \mathrm{P}^2) \tag{7}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are two input vectors and we can expand (7) in terms of matrix-vector multiplication $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\gamma \left[(\boldsymbol{x}_i \cdot \boldsymbol{x}_i^{\mathrm{T}} + \boldsymbol{x}_j \cdot \boldsymbol{x}_j^{\mathrm{T}} - 2\boldsymbol{x}_i \cdot \boldsymbol{x}_j\right]\right)$. Thus, we can get

$$\boldsymbol{M}(i,j) = \exp\left(-\gamma \left[\boldsymbol{A}(i,\cdot)\cdot\boldsymbol{A}(i,\cdot)^{\mathrm{T}} + \boldsymbol{B}(\cdot,j)^{\mathrm{T}}\boldsymbol{B}(\cdot,j) - 2\boldsymbol{A}(i,\cdot)\boldsymbol{B}(\cdot,j)\right]\right) \tag{8}$$

where $\boldsymbol{M}$ is the kernel matrix, $\boldsymbol{A}$ and $\boldsymbol{B}$ are input vector matrixes. In this form, it can be easily processed by the CUDA basic linear algebra library and GPU helper function.

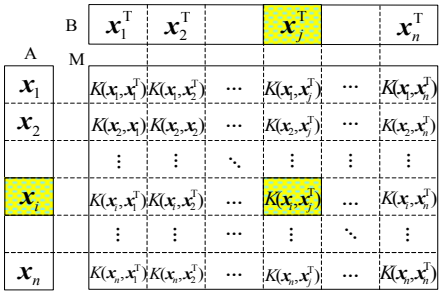

Figure 3. Structure of kernel matrix

The structure of KM is illustrated in Fig. 3, from which we can see the calculations of the elements in the kernel matrix are independent of each other, which is suitable for parallel computing.

Recently, GPUs have become the mainstream device for general-purpose computation because of their relatively low cost and high performance. CUDA is a convenient C-language programming API programming interface, which makes a sizable boost for the wide applications of GPUs in scientific computing areas [16]. In CUDA, CPU works as the host that mainly deals with the logic events and serial computation, while GPU works as the device that can execute multiple concurrent threads. Table 2 shows the pseudo code of the KM calculation process, from which we can see the pre-calculation is performed by combining CPU and GPU to get the best performance.

TABLE II.
PSEUDO CODE OF GPU-BASED KM CALCULATION

Kernel matrix calculation

1. Change the form of training vectors into column wise for GPU-based calculation.
2. Initialize the GPU equipment and allocate memory on GPU for the training vectors array.
3. Load the training vectors to the GPU memory.
4. Set parameters for kernel function and run it to call the calculation process on GPU. (Above four steps are performed on CPU side.)
5. For (each training vector) do
   - Load the training vector to the GPU.
   - Perform the calculation according to Equation 8.
   - Retrieve the calculation results from GPU.
6. End do
7. De-allocate GPU memory.

IV. IMPLEMENTATION ON WHEELED ROBOT

To verify the effectiveness of the proposed fast training method, implementation on a wheeled robot was carried out in this study. First, sensor data were sampled and preprocessed. Second, sample reduction was done based on the proposed SVS algorithm and a PCA model was established for fault feature extraction. Third, the GPU-based KM calculation was performed to reduce the training time. Finally, an SVM model was trained and the fault diagnosis task was accomplished. The flowchart of the experimental process is illustrated in Fig. 4.

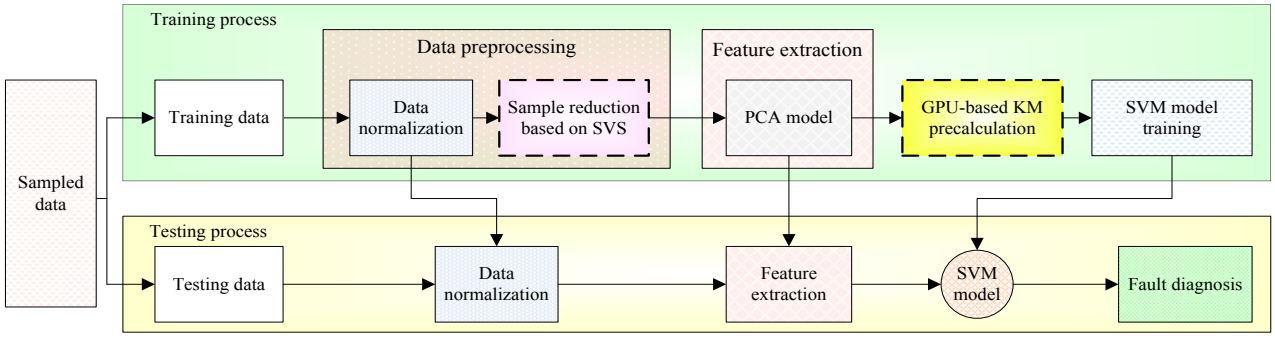

Figure 4. Flowchart of experimental process

A. Experimental Robot and Its Fault Diagnosis Problem

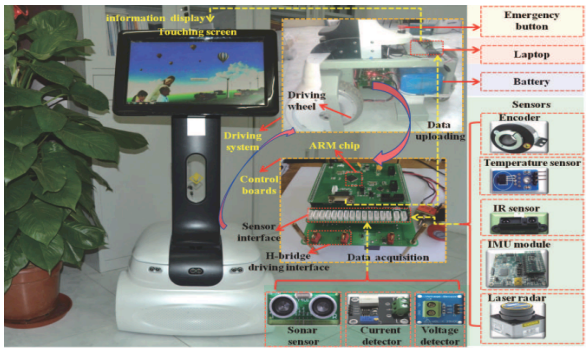

Figure 5. Experimental mobile robot

As shown in Fig. 5, the wheeled robot designed by our research group was applied as the experimental platform [17]. The robot was driven by two differential wheels and was equipped with various kinds of sensors, such as encoders, voltage detector, temperature sensor, etc. In this study, we mainly focused on the diagnosis of 7 common kinds of faults that occurred in the robot's driving system. As shown in Table 3, the fault space can be defined as $S_{err} = \{S_0, S_1, \mathrm{K}, S_7\}$, where S_0 represents the normal state and S_i represents the ith fault state.

TABLE III.
Fault categories TYPE

Fault categories	Fault position	Fault mode	Label
Normal condition	None	None	S_0
Mechanical faults	Left wheel	Low presure	S_1
	Right wheel	Low presure	S_2
	Left coupling	Loosening	S_3
	Right coupling	Loosening	S_4
Sensor faults	Left encoder	Pulse loss	S_5
	Right encoder	Pulse loss	S_6
	Gyroscope	Constant drift	S_7

B. Data sampling and preprocessing

In this study, 400 sets of data under each of the running states (S_0-S_7) were sampled respectively. Thus, the raw data sets can be denoted by $D_{all} = [D_0, \mathrm{K}, D_i, \mathrm{K}\ D_7]^{\mathrm{T}} \in R^{3200\times 9}$ and $D_i \in R^{400\times 9}$ represents the data set sampled under the ith fault condition. Then we randomly selected 200 sets of samples in each D_i as the training samples $X_i \in R^{200\times 9} (i = 0, \mathrm{K}, 7)$ and the rest 200 sets of samples were used as the testing samples $Y_i \in R^{200\times 9} (i = 0, \mathrm{K}, 7)$.

For a data set of 200 observations and 9 process variables $X_i \in R^{200\times 9}$, we can get the standardized data matrix $\bar{X}_i$ by Equation 9.

$$\bar{d}_{ij} = \frac{d_{ij} - \frac{1}{n}\sum_{i=1}^{n} d_{ij}}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(d_{ij} - \frac{1}{n}\sum_{i=1}^{n} d_{ij})^2}} \tag{9}$$

where d_{ij} is the element of matrix X_i and $\bar{d}_{ij}$ is the element of the standardized data matrix $\bar{X}_i$.

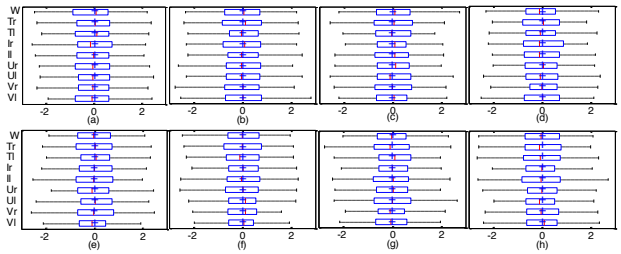

Figure 6. Boxplot of standardized data matrix (a) $\bar{x}_0$ (b) $\bar{x}_1$ (c) $\bar{x}_2$ (d) $\bar{x}_3$ (e) $\bar{x}_4$ (f) $\bar{x}_5$ (g) $\bar{x}_6$ (h) $\bar{x}_7$.

Fig. 6 demonstrates the boxplot of the standardized data matrixes from which we can see that the mean and variance of each dimension of $\bar{X}_i$ equal 0 and 1, respectively, after standardization. Then the proposed SVS algorithm (see Section III) was carried out to do sample reduction. In this study, c=100, k=9, and the final training data set $\bar{C}_T \in R^{976\times 9}$.

C. Feature Extraction and SVM Training

PCA is one of the most widely used feature extraction methods [9] due to its simplicity and high performance. One of the key procedures of PCA is to determine the optimal number of principal components (PCs). In our experiment, we set the threshold value at $\eta_r = 0.85$ and get the number of PCs $l = 5$. The cumulation variance proportion of the PCs is illustrated in Fig. 7. The final training data set $\bar{C}_T \in R^{976\times 9}$ is projected onto the principal component subspace, and we get the feature vectors $F_l \in R^{976\times 5}$.

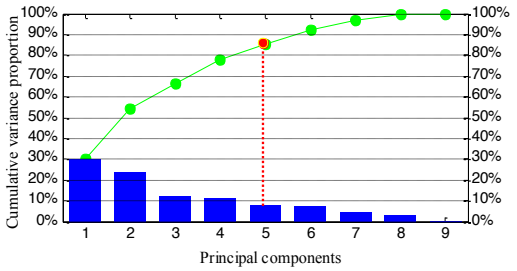

Figure7. Cumulation variance proportion

With the feature vectors $F_l \in R^{976\times 5}$, the SVM model is trained based on the procedures illustrated in Fig. 2. During the training process, a grid-search method was applied to optimize the parameters of the SVM; the parameters tuning process is demonstrated in Fig. 8, from which we can see the variation range of the parameters is 2^{-8}~2^{8}. To reduce the training time, we pre-calculated the kernel matrix $M \in R^{976\times 976}$ based on the procedures shown in Table 2. On the one hand, the pre-calculation avoids re-calculation during the k-fold cross validation procedure; on the other hand, the application of GPU-based parallel computing accelerates the pre-calculation process. Finally, the training process was accomplished, and we found the optimal SVM model, which was used to classify 8 kinds of running conditions of the robot. The training time and accuracy can be seen in the first group of

the contrast experiments, which are elaborated in Section *D*.

D. Contrast Experiments

To verify the effectiveness of the proposed fast training method, 5 groups of contrast experiments were carried out between the traditional method and the proposed fast training method. In group 1, 200 samples were sampled under each of the running conditions (S_0-S_7) and the whole training datasets in group 1 can be denoted by $D_{\text{all}}^1 \in R^{1600\times 9}$. Similarly, we can get $D_{\text{all}}^2 \in R^{2000\times 9}$, $D_{\text{all}}^3 \in R^{2400\times 9}$, $D_{\text{all}}^4 \in R^{2800\times 9}$ and $D_{\text{all}}^5 \in R^{3200\times 9}$ for the other four groups of experiments.

For fair comparison, all experiments were implemented on the same experimental PC, which was equipped with a quad core Interl Core i5-240 processor with 8GB of DDR3 RAM and a NVIDIA GTX-750Ti GPU with 2GB on board RAM. In all experiments, a PCA algorithm was used for feature extraction and the grid-search method was applied for parameters tuning. The experimental results are shown in Table 4 and Fig. 9.

TABLE IV.
Experimental results

Group	Method	Number of original training samples	Number of training samples after SVS	Number of support vectors	Training time (s)	Testing accuracy
1	Traditional	1600	--	445	436.3	89.4%
	Proposed	1600	976	411	53.2	88.5%
2	Traditional	2000	--	514	614.6	92.3%
	Proposed	2000	1060	483	60.1	91.6%
3	Traditional	2400	--	553	892.3	92.5%
	Proposed	2400	1320	526	71.3	92.0%
4	Traditional	2800	--	618	1184.8	90.2%
	Proposed	2800	1512	599	79.0	90.0%
5	Traditional	3200	--	855	1378.8	90.2%
	Proposed	3200	1792	846	85.1	89.8%

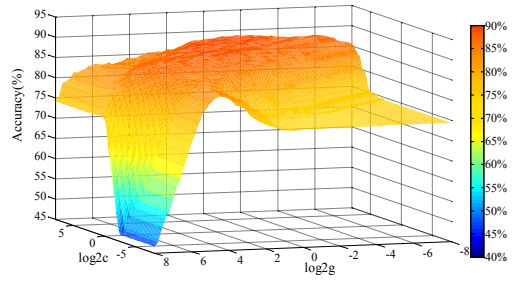

Figure 8. Parameters tunning for SVM

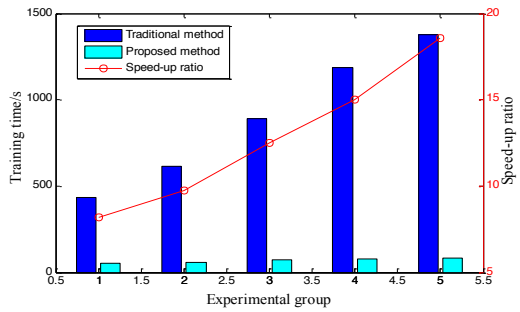

Figure 9. Speed-up ratio

From Table 4 and Fig. 9 we can see the training time of the traditional method increased rapidly with the increase of training samples. The speed-up ratios of the proposed method in Groups 1-5 are 8.2, 9.7, 12.5, 15.0 and 18.6, respectively, which implies that the more training samples provided, the more obvious the advantages of the proposed method will be. Moreover, as the proposed method retained most of the support vectors, there was no apparent decrease in the diagnosis accuracy.

V. CONCLUSION AND FUTURE WORK

In this paper, a novel fast training method for SVM was proposed. To increase the training speed, on the one hand, sample reduction was done based on the proposed support vectors selection algorithm, which could retain most of the support vectors and ensure good classification accuracy. On the other hand, we took advantage of the parallel computing abilities of GPU to pre-calculate the kernel matrix, which avoided recalculation during the cross validation process. To verify the effectiveness of the proposed method, an application in robot fault diagnosis is provided in detail. PCA is used to do feature extraction, and a grid-search method combined with 5-fold cross validation was applied to optimize the parameters of the SVM. Five groups of contrast experiments were carried out, and experimental results illustrated the effectiveness of the proposed method.

For future work, we would like to use the proposed method to achieve online fault diagnosis.

ACKNOWLEDGMENT

The authors would like to thank all the Editors and the anonymous reviewers for their valuable comments which are helpful for us to improve this paper.

REFERENCES

[1] W. Li, H. Xiong, H. Sun, Y. Mao, and H. Zhang, "De-noise of Online Monitoring Basic Data Collected by Surveying Robots," *International Journal of Online Engineering (iJOE)*, vol. 11, no. 2, pp. 68-72, 2015. http://dx.doi.org/10.3991/ijoe.v11i2.4476

[2] P. Chanak, I. Banerjee, J. Wang, and R.-S. Sherratt, "Obstacle avoidance routing scheme through optimal sink movement for home monitoring and mobile robotic consumer devices," *IEEE*

Transactions on Consumer Electronics, vol. 60, no. 4, pp. 596-604, 2014. http://dx.doi.org/10.1109/TCE.2014.7027292

[3] A. Maiti and S. Mahata, "Internet-Based Robot Assisted RF and Wireless Laboratory for Engineering Education," *International Journal of Online Engineering (iJOE)*, vol. 8, no. 7, pp. 28-33, 2012.

[4] F. Caccavale, A. Marino, G. Muscio, and. F. Pierri, "Discrete-time framework for fault diagnosis in robotic manipulators," *IEEE Transactions on Control Systems Technology*, vol. 21, no.5, pp. 1858-1873, 2013. http://dx.doi.org/10.1109/TCST.2012.2212196

[5] L. Yu, M. Wu, Z. Cai, and Y. Cao, "A particle filter and SVM integration framework for fault-proneness prediction in robot dead reckoning system," *WSEAS Transactions on Systems*, vol. 10, no. 11, pp. 363-375, 2011.

[6] N.-B. Hoang and H.-J. Kang, "A model-based fault diagnosis scheme for wheeled mobile robots," *International Journal of Control, Automation and Systems*, vol. 12, no. 3, pp. 637-651, 2014. http://dx.doi.org/10.1007/s12555-013-0012-1

[7] Y. Xu, Y.-J. Chen, and Q.-X. Zhu, "An Extension Sample Classification‐Based Extreme Learning Machine Ensemble Method for Process Fault Diagnosis," *Chemical Engineering & Technology*, vol. 37, no. 6, pp. 911-918, 2014. http://dx.doi.org/10.1002/ceat.201300622

[8] G. Moser, P. Costamagna, A.-D. Giorgi, A. Greco, L. Magistri, L. Pellaco, and A. Trucco, "Joint Feature and Model Selection for SVM Fault Diagnosis in Solid Oxide Fuel Cell Systems," *Mathematical Problems in Engineering*, vol. 2015, Article ID 282574, 12 pages, 2015.

[9] X. Yuan, M. Song, F. Zhou, and Z. Chen, "Fault diagnosis of service robot based on multi-PCA models and SVM-DS fusion decision," *Journal of Vibration, Measurement and Diagnosis*, vol. 35, no.3, pp. 434-440, 2015.

[10] L. Guo and S. Boukir, "Fast data selection for SVM training using ensemble margin," *Pattern Recognition Letters*, vol. 51, pp. 112-119, 2015. http://dx.doi.org/10.1016/j.patrec.2014.08.003

[11] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, and V. Vapnik, "Parallel support vector machines: The cascade svm," in *Advances in Neural Information Processing Systems*. MIT Press, 2005, pp. 521-528.

[12] Z. Fang, W. Junfang, and S. Wenbo, "SVM Fast Classification Algorithm of Based on Similarity Analysis," *International Journal of Digital Content Technology and its Applications*, vol. 7, no. 2, pp. 10-16, 2013.

[13] D. Tao, X. Tang, X. Li, and X. Wu, "Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1088-1099, 2006. http://dx.doi.org/10.1109/TPAMI.2006.134

[14] Q. Li, R. Salman, E. Test, R. Strack, and V. Kecman, "Parallel multitask cross validation for Support Vector Machine using GPU," *Journal of Parallel and Distributed Computing*, vol.73, no. 3, pp. 293-302, 2013. http://dx.doi.org/10.1016/j.jpdc.2012.02.011

[15] K. Sopyla and P. Drozda, "GPU Accelerated SVM with Sparse Sliced EllR-T Matrix Format," *International Journal on Artificial Intelligence Tools*, vol. 24, no. 1, pp.1-12, 2015. http://dx.doi.org/10.1142/S0218213014500122

[16] A. Athanasopoulos, A. Dimou, V. Mezaris, and I. Kompatsiaris, "GPU acceleration for support vector machines," *WIAMIS 2011: 12th International Workshop on Image Analysis for Multimedia Interactive Services,* Delft, The Netherlands, April 13-15, 2011.

[17] X. Yuan, M. Song, F. Zhou, Z. Chen, and Y. Li, "A Novel Mittag-Leffler Kernel Based Hybrid Fault Diagnosis Method for Wheeled Robot Driving System," *Computational Intelligence and Neuroscience*, vol. 2015, Article ID 606734, 11 pages, 2015.

AUTHORS

Xianfeng Yuan is currently pursuing his Ph.D. in the School of Control Science and Engineering, Shandong University, Jinan 250061, China. He received a B.S. from Shandong University, in 2011. His research interests include service robot, machine learning and intelligent fault diagnosis (e-mail: yuanxianfeng_sdu@126.com).

Fengyu Zhou,* corresponding author, is a professor in the School of Control Science and Engineering, Shandong University, Jinan 250061, China. His research interests include robotics, intelligent control and fault diagnosis (e-mail: zhoufengyu@sdu.edu.cn).

Mumin Song is with School of Control Science and Engineering, Shandong University, Jinan 250061, China.

Yugang Wang is with School of Control Science and Engineering, Shandong University, Jinan 250061, China.

Zhumin Chen is with School of Computer Science and Technology, Shandong University, Jinan 250101, China.

This work is supported by the National Natural Science Foundation of China (No. 61375084), the Key Program of Natural Science Foundation of Shandong Province (No. ZR2015QZ08) and the Fundamental Research Funds of Shandong University (No. 2014JC034). Submitted 07 July 2015. Published as resubmitted by the authors 25 October 2015.