International Journal of Knowledge and Language Processing
Volume **2**, Number 4, October **2011**

www.ijklp.org
KLP International ©2011 ISSN 2191-2734
pp. **28-38**

Incremental SVM Learning with Agent-based Framework on Data Stream

Zhongwei Li
[1]Nankai University
No.94, Weijin Road, Nankai District, Tianjin, China
lizhongwei@nankai.edu.cn

Received September 2011; revised October 2011

ABSTRACT. *Support Vector Machine (SVM) is a statistical method with excellent classification performance based on small-sample statistics. It has become a popular research topic among specialists in recent years that how to apply SVM's excellent learning ability to large-scale database, especially to data stream. This paper makes an introduction for the basic principles of SVM and an analysis of the QP problem. And analysis is made for application of SVM learning method structure in the large datasets with crossed feedback to improve the performance of the typical structure of incremental SVM method. Finally, combined with Multi-agent system framework, a framework that supports incremental SVM learning method and extendable Multi-agent system on data stream is proposed in this paper. According to the related experiment and analysis of the results, this frame is proved efficient.*

Keywords: SVM, Incremental learning, Multi-agent framework, Data stream

1. **Introduction.** Support Vector Machine (SVM) is a powerful classification and regression tool and has been applied to various pattern classification problems successfully in recent years, but its computation and storage requirements increase rapidly with the number of training vectors. The reason is that the core problem of an SVM is a quadratic programming problem (QP), separating support vectors from the rest of the training data [1]. General-purposed QP solvers tend to scale with the cube of the number of training vectors. Therefore, how to efficiently train support vector machines on problems with a large training dataset is still an ongoing research issue.

One approach for accelerating the QP is based on 'chunking' technology where subsets of the training data are optimized iteratively [2], until the global optimum is reached. Eliminating non-support vectors early during the optimization process is another strategy that provides substantial savings in computation. Parallel training methods have been proposed and proved that they are more suitable and efficient for SVM by splitting the problem into smaller subsets and training a network to assign samples of different subsets, and multiple SVM classifiers are trained in parallel way or in distributed computer system [3]. A. Tveit et al proposed the cascade SVM structure and proved efficient[4], and other

researchers improved this structure to obtain more satisfying results [5, 6]. But to improve that training speed through parallelization has obvious difficulty due to dependencies between the computation steps and the distribution status of training data in every subset.

The basic principle of SVM is to find an optimal separating hyperplane so as to separate two classed of patterns with maximal margin. It tries to find the optimal hyperplane making expected errors minimized to the unknown test data, while the location of the separating hyperplane is specified via only data that lie close to the decision boundary between the two classes, which are support vectors [1]. Obviously, the design of SVM allows the number of support vectors to be small compared to the total number of training data, therefore, SVM seem well suited to be trained according to incremental learning.

Multi-agent technology, which has been rigorously researched since early 1990s, is being regarded as one of the promising technologies for enterprise complex software systems [7]. The features of Multi-agent technology such as autonomy, distributed collaboration, and intelligence naturally fit with the characteristics of large-scale database. Multi-agent system, means that agents can work together or act autonomously in their environment to complete a set of goals. Now, more and more researches upon it are carrying on. Ref. [7] applied the Multi-agent for energy supply and demand prediction. Ref.[8] proposed a framework for remote diagnostics with Multi-agent. More researches are carried out in fields of e-learning, intelligent tutoring system, supply chain system, etc.

For the characteristic of Multi-agent system, the idea that constructs the distributed mining framework with Multi-agent system for incremental SVM learning on large-scale database is proposed in this paper. In this framework, every agent carries the local training data and the distribution status. When an agent gets a local result by the local SVM training algorithm, it will compare its distribution with the ones of the other agents. If the difference is not clear, the local result could be discarded, else the data of every part should be changed according to their distribution status.

This paper is organized as follows. In section 2, brief introduction is made for the SVM and the QP problem. In section 3, this paper introduced the typical structuring method for incremental SVM learning and its improvement method, with the analysis of the key parts in the improved method. In section 4, combined with Multi-agent system framework, a framework based on incremental SVM training structure is proposed. In the end, the conclusions are outlined.

2. **SVM and QP Problem.** Classification is achieved by realizing a linear or non-linear separation surface in the input space by SVM [1]. The aim of SVM is to find optimal hyperplane between two classes with the symbols $y \in \{1,-1\}$, and y_i is label of the ith training sample, $x_i \in R^n$. For a classification problem, given l samples data points $\{(x_1, y_1),(x_2, y_2),\cdots,(x_l, y_l)\}$, SVM training involves solving a quadratic programming problem and the optimal solution gives rise to a decision function of the following form:

$$f(x) = \mathrm{sgn}[\sum_{i=1}^{l} y_i a_i (x \cdot x_i) + b] \tag{1}$$

Often, only small fractions of a_i coefficient (Lagrange multipliers) are non-zero, corresponding samples are called support vectors. Training a SVM on the support vectors alone gives the same result as training on the whole example set, therefore, remaining samples may be regarded as redundant and negligible because they do not contribute to the decision function.

When training samples are non-separable, one can transform the set of input samples into a higher dimensional feature space using a map $\Phi(x_i) \mapsto z_i$, and then execute a linear separation. This leads to:

$$f(x) = \text{sgn}[\sum_{i=1}^{l} y_i a_i (\Phi(x) \cdot \Phi(x_i)) + b] = \text{sgn}[\sum_{i=1}^{l} y_i a_i K(x, x_i) + b] \quad (2)$$

$K(x, x_i) = (\Phi(x) \cdot \Phi(x_i))$ is called the kernel function.

The kernel function allows us to construct an optimal separating hyperplane in a new feature space without explicitly performing calculations. Unfortunately, the training of Support Vector Machine itself and kernel function can be very time and memory consuming, for large amounts of data and kernel matrix will be stored and computed. The reason is training a SVM consists in finding a that minimizes the objective function:

$$Q(a) = -\sum_{i=1}^{n} a_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j y_i y_j K(x_i, x_j) \quad (3)$$

Subject to the constraints:

$$\sum_{i=1}^{n} a_i y_i = 0 \qquad 0 \le a_i \le C \quad (4)$$

That is to say, a quadratic optimization problem need be solved, where the number of parameters is n. This makes the use of SVM for large dataset difficult: computing $K(x_i, x_j)$ for every training data pair would require $O(n^2)$ computation, and solving may take up to $O(n^3)$. However, current state-of-the-art algorithms appear to have training time complexity scaling much closer to $O(n^2)$ than $O(n^3)$ [2].

3. **Incremental Learning on Large-scale Database.** The development of modern computing and information technologies has enabled that huge amount of information has been produced as digital data format. It is impossible to classify this information with hand one by one in many realistic problems and fields, there is a need to scale up inductive learning algorithms to handle more training data. A problem of SVM is the training datasets are common far too large or always change in practice and new samples data are added in at any moment, and incremental learning for SVM should be developed to avoid running time-consuming training process frequently. Many researches have been made on incremental learning with SVM [9, 10].

As the location of the optimal hyperplane is only related with linear combination of support vectors, most incremental learning algorithms sets focus on collecting more useful training data as support vectors with batch training model. Given that only a small fraction of training data end up as support vectors, the SVM is able to summarize the data space in a

very concise manner. It is a feasible method that we can partition the training dataset in batches that fit into memory, for each new batch, a SVM is trained on this batch and the support vectors from the previous training step[9], as the Figure 1 shown.

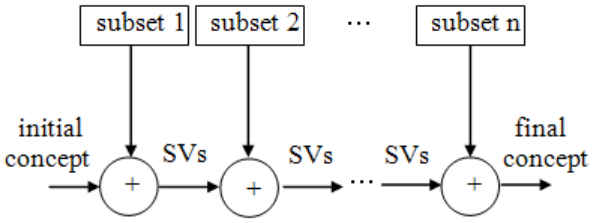

FIGURE 1. Batch incremental SVM learning model.

Using this method, the learning results are "incremental" combined and deposited. The batch learning methods utilize the property of SVM that only a small fraction of training data end up as support vectors, the SVM is able to summarize the data space in a very concise manner, and assume that the batches of data will be appropriate samples of the data. Clearly, the problem is the learning results are subject to numbers of batches and state of data distribution but always the distribution of data is unknown. That is to say, the problem is the learning results are subject to numbers of batches and state of data distribution. According to important properties of support vectors, we can expect to get an incremental result that is equal to the non-incremental result, if the last training set contains all samples that are support vectors in the non-incremental case. But disadvantage of this learning model is the time consuming is not prompted, since all data in any subset, should be computed to tell whether it is a support vector or not.

Clearly, it is confirmed that only the storage need can be decomposed in batch training way. To speed up the incremental training process, some parallel training algorithms with multiple SVM classifiers are developed and proved that they are more suitable to learn on large-scale dataset than a single SVM classifier [11-13]. Therefore, taking advantage of multiple SVM classifiers in batch training model is an improvement to incremental learning techniques with SVM, and could expect to obtain more satisfying training results to deal with large-scale classification problems.

In true incremental learning, the training dataset is not fully available at the beginning of the learning process as in batch learning, data can arrive at any time, so incremental learning algorithms differs from batch ones greatly. It is proved that the location of the optimal hyperplane is only related with linear combination of support vectors. This implies that the key to construct optimal hyperplane is collection more useful data point as support vectors during the incremental learning. Most incremental learning algorithm are based on improving SVM training process by collecting more useful data as support vectors [10, 14].

On the other hand, eliminating non-support vectors early from the optimization is an effective strategy for speeding up training SVM [15]. From this concept, the cascade structure is developed by initializing the problem with a number of independent smaller optimizations and the partial results are combined in later stages in a hierarchical way, as

shown in figure 2, supposing the training data subsets TD_1, TD_2, TD_3 and TD_4 are independent among each other[16].

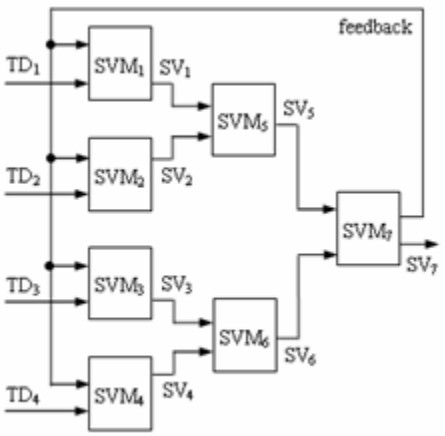

FIGURE 2. Cascade SVM structure.

In the structure shown in figure 2, splitting the training data and combining the results can be done by many different ways. Sets of support vectors from two SVM classifiers are combined and the optimization proceeds by finding the support vectors in each of the combined subsets, and this continues until only one set of vectors is left. The advantage is that every SVM did not have to deal with the whole training dataset, and these multiple SVM classifiers can be trained in distribute computer network, so the training process is speeded up greatly. Often, this cascade structure produces satisfactory accuracy with a single pass through, but if the global optimum has to be reached, the result of the last layer should be fed back into the first layer. Therefore, we should consider when a feedback is needed and how to collect support vectors efficiently.

In the structure shown in figure 2, 7 SVM classifiers are constructed in total. To speed up the whole training process, B. L. Lu et al proposed parallel training methods based on cascade structure, ignoring the feedbacks which are necessary and combined the SV_5 and the SV_6 into SV_7, even combined SV_1, SV_2, SV_3 and SV_4 directly in revised algorithms[17].

In fact, considering the distribution of training dataset and the need of global optimization, feedback should be kept to adjust classifiers. Therefore, we improved the cascade structure in the following aspects:

1. If needed, take SV_5 and SV_6 as the feedback input directly.

2. The feedback is not added into each subset in the first layer directly but in a crossed way, that is, SV_5 is added into TD_3 and TD_4, while SV_6 into TD_1 and TD_2.

3. Use the standard SVM algorithm, but collect more potential data samples as the support vectors.

The key to construct a SVM is to obtain support vectors, and the standard SVM algorithm obtains them by computing and judging whether $a_i = 0$ or not. Domeniconi. et

al [11] compared four methods to collect samples as support vectors in incremental learning and proposed algorithm applied to large-scale stream data. However, the fixed partition and the exceeding margin techniques are limited to some extent, because the size of partition lies on the dimension and type of training datasets, and the data samples near the margin defined by SVM are also sensitive to hyperplane, while given a fixed margin. This often lies on the experiments greatly. Here, we proposes to compute the mean distance of positive and negative training data to the hyperplane respectively and then collect training data of which distance to the hyperplane is less than the corresponding mean distance as support vectors.

That is, if the distance of one data to hyperplane is less than the value of all data samples' distance mean, it should be collected as a support vector and added to the training dataset to update the SVM classifiers which need to be adjusted. The value of all data samples' distance mean can be calculated by summing up the amount of the positive data samples and negative ones. The reason behind our method is that the data samples' distance mean could reflect the statistical meaning of samples' distribution compared with these traditional methods.

The feedback is necessary if the results are not satisfying at the first learning process, because that the distribution status will change the position of the final hyperplane. Therefore, we adopt feedback to overcome the potential influence by adjusting our multiple SVM classifiers. Generally, we can judge whether there need feedback or not by given a ε according to field knowledge by experts. If the updated value of every SVM is less than this given ε, the feedback is considered. But there is a simpler and more convenient way, which is to compare the difference of the current feedback and the previous one. If the difference set of these two feedbacks is empty or fixed or less than a given number e, the feedback is not needed any more.

The experiment results and the analysis are given based on this arithmetic in Ref. [16], which proves the efficiency of the delta training method.

4. **Learning Framework based on Multi-agents Framework.** Data stream is a kind of large-scale database and its distinct characteristic is that the data streams vary with the time. The typical data streams may be obtained in stock trade, credit card consumption, weather information, and so on. So, how to generalize the incremental SVM learning methods to data streams, a special kind of database, is one challenging task.

Distributed mining framework is constructed according to the idea of "Divide and Conquer". Supposed that at every time point T, the data on every distributed node are fixed. They will be updated till next time point T+1. The easiest way is training data on every node at time point T. As the time passed, there will be serials of learning results data on every node. Then by some rules or fusion expert knowledge, some local learning results data will be selected as the global ones.

If the distribution of data on every node is balanced, or the same, or not varied with time T and T+1, the above method is always right. But this is a hypothetical instance. That is to say, the features of collaboration between agents are used to make the distribution of data

balanced. Thus, the local learning results data could be the global ones.

The structure of the proposed incremental SVM learning framework based on Multi-agent system is shown in figure 3. There are 4 main kinds of agent in the structure. Each agent has its own function.

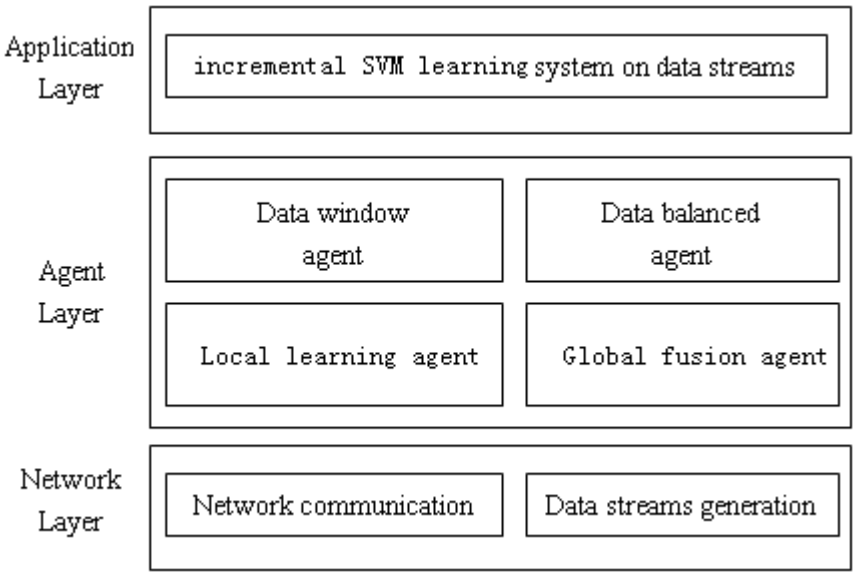

FIGURE 3. Structure of Multi-agent system framework.

(1) Data windows agent

The main function of data window agent is to separate the data at time point T into some smaller ones. The width of data window is the max value n that local outlier data mining algorithm can deal at time point T. Different local SVM learning algorithm has different capability on n data between T and T+1. If different local SVM learning algorithms are assigned to distributed nodes, the data window agents will be different.

(2) Data balanced agent

Data balanced agent will check the distribution status on each node at time point T and compare them with that of time point T+1 according the density of data. If the difference is under the threshold value pre-given, the data cannot be updated and the local SVM learning algorithm will not be executed at the time point T+1. If the difference of density of data at the T and T+1 is obvious, the data on this node will be updated and the local SVM learning algorithm will be executed again to find the support vector data at T+1.

(3) Local learning agent

The function of local learning agent is to execute the local SVM learning algorithm according the message from the data balanced agent, and record the local learning results data at different time point T.

(4) Global fusion agent

When all data are done, the global fusion agent will select appropriate learning results data as the global ones. Of course, this is a decision-making process.

5. **Application Experiment.** With the development of society and the need of modernization, the City Operation System has accumulated lots of useful data, and it has become a focal point in this field that how to find knowledge and make a strategic decision. Compared with available mass monitoring data, the potential symptoms of accidents are "small samples" information models. How to find these symptoms in the mass monitoring data of the City Operation System is crucial and important to prevent the accidents happening. To validate feasibility of proposed learning model, here combining the method of outlier data mining with SVM, it is applied to mine potential symptoms in unforeseen accidents of the City Operation System.

There are mass monitoring data in the City Operation System, which are collected from different departments and fields of the city, such as traffic, weather, all kinds of water, electricity, windpipe line, and etc. The relations between these data are complicated; the change of some data may have direct or indirect influence on the other data. Among the mass and complicated data, the data which have great influence on the normal running of the City Operation System should be given due attention. However, the change of some data are not obvious, but the data still have some impact to the city system's normal running, that are the outlier data of the City Operation System. The traditional data forecast methods can describe and analyze the change trend of all kinds of data easily, but it can't give a quantitative express of the degree of the relation between the data. Therefore, the outlier mining of the City Operation System is obviously important.

The format of data used to simulation analyze in this application are listed in Table.1. The more data are not listed for some reasons. All the data to be analyzed should be combined into one table, according the monitor date.

TABLE 1. Some data of water monitoring

date	**Water, electric, gas**					
	Water supply	**Plan amount of water**	**Electric supply**	**Plan amount of electric**	**Gas supply**	**Plan amount of gas**
20111109	390	383	1785.9	1900	1193.4	1100
20111110	399	337	1832.3	1850	1232.7	1400
20111111	386	358	1799.8	1900	1366.9	1500
20111112	370	353	1815.4	1900	1284.1	1500
……	……	……	……	……	……	……

We conducted experiment using these data, which is pre-labeled by hand and consists of 5879 data points, each having a dimension of 57 to compare the batch incremental SVM learning algorithm and crossed iteration incremental SVM learning algorithm we proposed. The experiment is prepared as following: take 477 data points as initial training set and the other data points as test set randomly, separate the rest data points into 5 subsets and use the polynomial kernel.

The application process proposed in this paper is as follows: first, some necessary

parameters are set in the user interface program. The data window agents are assigned to each distributed node to prepare the mining data when the time point T starts, and local learning agents are working. When it finished, all local learning data on each node will be record. Then the T+1 comes, the data balanced agents do their work that is to compare the data distribution status at the two time points. If there are no differences in distribution, or within the threshold value, the data streams at time point T+1 are discarded immediately and the data windows agents are ready to deal the next data set at T+2. Else, the learning results data, those are support vectors, obtained at T will be added to the data set at T+1, and then the local learning agents do their work at the T+1. Repeat this process until the last time point T or all data streams are done, there is a set of outlier data on each distributed node of mining framework.

At last, global fusion agents have their function. And the learning results are given in form of rules, and more clearer and understandable interpretations are given with visual graphic display.

Table 2 shows the classification precision comparisons for this experiment.

TABLE 2. Classification precision comparisons

Data number	**Agent number**	**Precision of Batch algorithm**	**Precision of proposed algorithm**
583	5	89.6%	87.8%
899	10	90.3%	91.7%
1275	15	91.5%	93.0%
2598	20	92.3%	93.8%
3676	20	92.8%	92.8%
4469	25	90.7%	92.1%
5366	25	88.3%	90.9%
5879	30	86.8%	90.7%

It can be seen that the classification precision results in incremental steps are improved. The whole classification process does not decrease the time-cost of computing SVM, but the classification precision is improved. Another interesting thing is when agent number increased in some decent, the precision results may be decrease. Support the data is standard distribution, why the precisions decrease? We think that with the agent increase, the time-cost of algorithm must decrease. How one agent gets relationship to the other ones? How the support vectors change with the number of agent and data increase? These are the next research points, but it is clear, the algorithm improves the classification precision compared to the Batch learning algorithm.

5. **Conclusions.** In this paper, an incremental SVM learning algorithm on data stream is introduced, which is based on the cascade structure. The problem can be decomposed into some smaller-sized training datasets and the corresponding multiple SVM classifiers

trained in parallel computer system or distributed structure system with necessary feedbacks. The operations on feedback are the important improvements of the batch learning model.

Multi-agent system has the features of autonomy, distributed collaboration and intelligence. These meets the needs of distributed mining framework construction to learning on data stream. Therefore, the Multi-agent system and the incremental SVM learning model are combined in this paper, proposed an incremental SVM learning framework for data streams. The proposed framework has been applied to analyze the symptoms in the mass monitoring data of the City Operation System practically.

Acknowledgment. This work is partially supported by "the Fundamental Research Funds for the Central Universities". The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[2] T. Joachims, *Making Large-scale Support Vector Machine Learning Practical*, Advances in Kernel Methods, Cambridge, MIT Press, 1999.

[3] R. Collobert, Y. Bengio, S. Bengio, *A Parallel Mixture of SVMs for Very Large Scale Problems*, Neural Information Processing Systems, Vol.17, MIT Press, 2004.

[4] A. Tveit, H. Engum, *Parallelization of the Incremental Proximal Support Vector Machine Classifier using a Heap-based Tree Topology*, Tech. Report, IDI, NTNU, Trondheim, 2003.

[5] J. X. Dong, A. Krzyzak, C.Y. Suen, A fast parallel optimization for training support vector machine, *Proc. of 3rd International Conference on Machine Learning and Data Mining*, Leipzig, Germany, LNAI 2374, pp.96-105, 2003.

[6] G. zanghirati, L. Zanni, A parallel solver for large quadratic programs in training support vector machine, *Parallel computing*, Vol.29, 2003.

[7] Y. G. Peng, Z. C. Lu; J. S. Yu. Multi-agent framework for engergy supply/demand prediction. *Proc. of 2009 World Congress on Computer Science and Information Engineering*, 2009, pp.586-590.

[8] Danil Prokhorov. Multi-agent framework for remote diagnostics. *Proc. of 2010 IEEE Congress on Evolutionary Computation*, 2010.

[9] N. Syed, H. Liu, and K. Sung, Incremental learning with support vector machines, *Proc. of IJCAI Conference*, Sweden, 1999.

[10] P. Mitra, C. A. Murthy, and S. K. Pal, Data condensation in large databases by incremental learning with support vector machines, *Proc. of ICPR Conference*, Spain, 2000.

[11] A. Tveit, M. L. Hetland, H. Engun, Incremental and decremental proximal support vector classification using decay coefficients, *Proc. of the 5th International Conference on Data Warehousing and Knowledge Discovery*. Lecture Notes in Artificial Intelligence, Spring-Verlag, 2003.

[12] R. Klinkenberg, T. Joachims, Detecting concept drift with support vector machines, *Proc. of the 17th International Conference on Machine Learning*, Morgan Kaufmann, 2000.

[13] K. Crammer., Y. Singer., On the learnability and design of output codes for multiclass problems,

Computational Learning Theory, pp.35-46, 2000.

[14] G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning, *Advances in Neural Information Processing Systems*, 2000.

[15] J. C. Platt, *Fast training of support vector machines using sequential minimal optimization*, Advances in Kernel Methods, Cambridge, MIT Press, 1998.

[16] J. P. Zhang, Z. W. Li, J. Yang. A parallel SVM training algorithm on large-scale classification problems. *Proc. of 2005 International Conference on Machine Learning and Cybernetics*, 2005, pp.1637-1641.

[17] Y. M. Wen and B. L. Lu, A cascade method for reducing training time and the number of support vectors, *Proc. of International Symposium on Neural Network*, Dalian, 2004.