Incremental Learning of Support Vector Machines by Classifier Combining

Yi-Min Wen[1,2] and Bao-Liang Lu[1]*

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, 800 Dong Chuan Road, Shanghai 200240, China
{wenyimin; bllu}@sjtu.edu.cn
[2] Hunan Industry Polytechnic, Changsha 410007, China

Abstract. How to acquire new knowledge from new added training data while retaining the knowledge learned before is an important problem for incremental learning. In order to handle this problem, we propose a novel algorithm that enables support vector machines to accommodate new data, including samples that correspond to previously unseen classes, while it retains previously acquired knowledge. Furthermore, our new algorithm does not require access to previously used data during subsequent incremental learning sessions. The proposed algorithm trains a support vector machine that can output posterior probability information once an incremental batch training data is acquired. The outputs of all the resulting support vector machines are simply combined by averaging. Experiments are carried out on three benchmark datasets as well as a real world text categorization task. The experimental results indicate that the proposed algorithm is superior to the traditional incremental learning algorithm, Learn++. Due to the simplicity of the proposed algorithm, it can be used more effectively in practice.

1 Introduction

The brain of human beings has powerful ability of incremental learning. Therefore, how to develop brain-like computing model, how to implement incremental learning is one challenge problem in machine learning research. In real world applications, there are three scenarios need incremental learning: all training data cannot be gathered at one time for the cost of collecting data. As a result the data are acquired batch by batch; some real world applications need instant learning once some training data obtained; all training data cannot be loaded into the memory of computers if the training set is very large. According to Jantke [1], incremental learning is to construct new hypothesis by using only the hypothesis before and the recent information on hand. Zhou and Chen [2] distinguished three kinds of incremental learning tasks: Example-incremental learning (E-IL);

* To whome correspondence should be addressed. This work was supported by the National Natural Science Foundation of China under the grants NSFC 60375022 and NSFC 60473040.

Class-incremental learning (C-IL); and Attribute-incremental learning (A-IL). However, C-IL and A-IL have not been received much attention so far. Syed *et al.* [3] introduced two types of incremental learning methdos: instance learning, which uses one example at a time, and block by block learning, which uses a suitable-size subset of samples at a time. Polikar *et al.* [4] defined a criteria of incremental learning algorithm as follows:

1. It should be able to learn additional information from new data.
2. It should not require access to the original data used to train the existing classifier.
3. It should preserve previously acquired knowledge (that is, it should be not suffer from catastrophic forgetting).
4. It should be able to accommodate new classes introduced with new data.

Syed *et al.* early started the work of incremental learning of support vector machines (SVMs) [5]. Their proposed algorithm preserves only support vectors at each incremental step and add them to the training set for the next step. Based on the work of Syed *et al.*, Dominiconi and Gunopulos made a further research on it [6]. They explored four different techniques for SVMs incremental learning: Error-driven technique, Fixed-partition technique, Exceeding-margin technique, and Exceeding-margin+errors technique. Cauwenberghs and Poggio [7] proposed an algorithm of SVMs online learning. In their algorithm, adiabatic increments retain the Kuhn-Tucker conditions on all previously seen training data once a new sample acquired. Diehl and Cauwenberghs expanded this algorithm to incrementally accommodate many samples at a time [8]. However, these algorithms should keep all the training data gathered so far to scan. Ralaivola and colleague proposed a local strategies for SVMs online learning [9]. Liu *et al.* [10] explored incremental batch learning with SVMs and showed that the incremental batch learning is to solve a convex quadratic programming the same as the standard SVMs algorithm.

At present, however, the essence of the training algorithms of various kinds of artificial learning systems is an optimization procedure that aims to ensure the generalization ability based on the current learning environment. Therefore, all the current machine learning algorithms don't adapt for incremental learning in nature. The non-adaption lies in that the computation model lacks the ability to get new knowledge or cannot retain the knowledge learned before [11]. The training of artificial neural networks is a gradient descent process, and therefore the modification of connection weights will damage the learned knowledge. The training of SVMs is a global optimization based on all training data. As a result, new added training data will make support vectors change. A lot of work has been done for handling this non-adaption problem. Artificial neural networks always get new knowledge and decrease forgetfulness by bounded weight adaptation or modifying the number of hidden units [12], however, the learning ability and knowledge capacity of neural networks will be limited. When there are concept drift in unseen training data, the algorithm proposed by Syed *et al.* will lost its availability [13].

Classifier combining is a useful method for machine learning [14] [15]. Classifier combining learning includes ensemble learning, modular learning, and meta learning. Many scholars have applied classifier combining techniques to incremental learning and various algorithms based on classifier combining have been proposed. Polikar *et al.* proposed Learn++ based on AdaBoost algorithm [4]. Learn++ satisfies the criteria of incremental learning. However, once a batch of incremental training data occurs, Learn++ should train many classifiers. Lu and Ichikawa proposed an incremental learning model based on emergence theory [16]. This computation model uses two emergence rules to integrate submodules [14], however, all the learned samples should be preserved. Macek proposed incremental learning algorithms based on bagging and boosting and successfully applied them to EEG data classification [17]. Wang *et al.* used weighted ensemble classifiers to mine concept-drifting data stream [18]. Like bagging, a model of incremental learning by classifier combining (ILbyCC) is proposed in this paper.

2 Incremental Learning by Classifier Combining

2.1 Definition of Batch Incremental Learning

The batch incremental learning problem can be defined as follows:

Definition 1. *: Given a sequence of training dataset* $S_1, S_2, ..., S_m$*, ,where* $S_i = \{(x_{i_j}, c_{i_j}) | x_{i_j} \in R^n, c_{i_j} \in L_i \subseteq \{1, 2, ..., k\}, 1 \leq j \leq n_i\}, 1 \leq i \leq m$*.* L_i *indicates the set of class label in training dataset* S_i*. Lets* E_1 *denotes the classifier trained on* S_1*, the batch incremental learning procedure* IL *can be illustrated as:* $IL(S_i, E_{i-1}) = E_i,\ 2 \leq i \leq m$*.*

In this paper, we only consider the case where the number of class labels don't decrease, i.e., $L_1 \subseteq L_2 \subseteq ... \subseteq L_m$. For convenience, three concepts are defined as follows:

1. Incremental batch—a new batch of training data.
2. Incremental step—an incremental learning on an incremental batch.
3. Incremental sequence—a sequence of incremental step.

For example, suppose there are six training datasets in a given sequence of incremental batch and their corresponding class label sets are like as $\{1, 2, 3\}$, $\{1, 2, 3\}$, $\{1, 2, 3\}$, $\{1, 2, 3, 4\}$, $\{1, 2, 3, 4\}$, and $\{1, 2, 3, 4\}$. Following the definations mentioned above, the incremental step is 6. These 6 incremental steps form an incremental sequence. According to the work [2], obviously, example-incremental learning takes place from the first to the third incremental steps, another example-incremental learning takes place from the fourth to the sixth incremental steps, while class-incremental learning takes place from the third to the fourth incremental steps.

ILbyCC takes a frame of modular architecture. Modular architecture can make classifier easy to be expanded, aiming to adapt to incremental learning. ILbyCC trains a new classifier on an incremental batch and save it. All the classifiers trained by far are combined into one combined classifier. The training algorithm of ILbyCC can be illustrated as: $M(f_1, f_2, ..., f_{i-1}, f_i) = E_i$, where, M denotes the strategy for classifier combining, E_i denotes the current combined classifier. Fig.1 illustrates the model of ILbyCC.

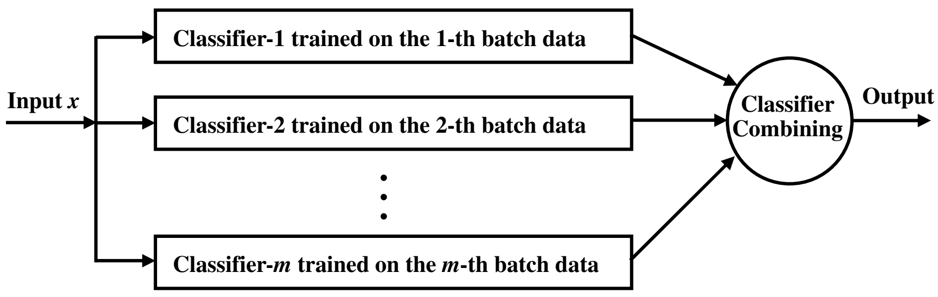
Input x
Classifier-1 trained on the 1-th batch data
Classifier-2 trained on the 2-th batch data
Classifier-m trained on the m-th batch data
Classifier Combining
Output

Fig. 1. The model of incremental learning by classifier combining

2.2 Combining Classifiers by Averaged Bayes

Classifier combining is an important strategy for pattern recognition. Various work has demonstrated that combining classifiers can promote generalization ability of single classier because of the complementarity between classifiers. Xu surveyed the field of classifier combining and proposed that classifiers combining can be implemented at three level: abstract level, rank level, and measurement level [19]. The Averaged Bayes used in this paper belongs to measurement level.

Given m classifiers that can output posterior probability information, when a test input x comes, the j-th classifier outputs the posterior probability of x belonging to all the classes:

$$P_j(y = i|x), i \in \{1, 2, ..., k\}, j = 1, 2, ..., m \tag{1}$$

According to Averaged Bayes, the combined classifier E_m computes the posterior probability of x belonging to all classes as follows:

$$P_{E_m}(y = i|x) = \frac{1}{m}\sum_{j=1}^{m} P_j(y = i|x),\ i \in \{1, 2, ..., k\} \tag{2}$$

According to Bayes rule, x can be classified as the i-th class:

$$i = \arg\ max_{i=1}^{i=k} P_{E_m}(y = i|x) \tag{3}$$

2.3 Incremental Learning Algorithm by Classifier Combining

Considering the difference between example-incremental learning and class-incremental learning, we can classifiy the training algorithms for ILbyCC into two types: example-incremental learning algorithm and class-incremental learning algorithm.

The example-incremental learning algorithm can be described as follows:

1. Input: given example-incremental learning sequence: $S_1, S_2, ..., S_m$, where, $S_i = \{(x_{i_j}, c_{i_j})|x_{i_j} \in R^n, c_{i_j} \in L_i \subseteq \{1, 2, ..., k\}, 1 \leq j \leq n_i\}, 1 \leq i \leq m, L_1 = L_2 = ... = L_m = L$.

2. For $t = 1, 2, ..., m$
   (a) Taking cross-validation on S_t to select the optimal parameters of training algorithm and train a classifier f_t on the incremental batch S_t.
   (b) Saving classifier f_t and S_t can be discarded.
3. Testing:
   (a) Import a test input x into each $f_t,\ 1 \le t \le m$, and calculate the posterior probability of x belonging to all the classes: $P_t^j, 1 \le t \le m, 1 \le j \le k$.
   (b) Take the rule of classifier combining M to a combination $f_t,\ 1 \le t \le m$, and get the combined classifier $E_m = M(f_1, f_2, ..., f_m)$. E_m outputs the posterior probability of x belonging to all the classes: $P_{E_m}^j, 1 \le j \le k$.
   (c) Classify x according to the value of arg$max_{j \in L} P_{E_m}^j$.
4. The algorithm ends.

The class-incremental learning algorithm can be described as follows:

1. Input: given two example-incremental learning sequences: $List_1 = \{S_1^1, S_1^2, ..., S_1^m\}$ and $List_2 = \{S_2^1, S_2^2, ..., S_2^n\}$. where, $L_1^1 = L_1^2 = ... = L_1^m = L1,\ L_2^1 = L_2^2 = ... = L_2^n = L2,\ L1 \subset L2$.
2. For $t = 1, 2, ..., m$
   (a) Take cross-validation on S_1^t to select the optimal parameters of training algorithm and train a classifier f_1^t on the incremental batch S_1^t.
   (b) Save classifier f_1^t and S_1^t can be discarded.
3. For $t = 1, 2, ..., n$
   (a) Take cross-validation on S_2^t to select the optimal parameters of training algorithm and train a classifier f_2^t on the incremental batch S_2^t.
   (b) Save classifier f_2^t and S_2^t can be discarded.
4. Testing:
   (a) Import a test input x into each $f_2^t,\ 1 \le t \le n$, and calculate the posterior probability of x belonging to all classes: $P_t^j, 1 \le t \le n, j \in L2$.
   (b) Take the rule of classifier combining M to combine classifiers $f_2^t,\ 1 \le t \le n$, and get the combined classifier $E_n = M(f_2^1, f_2^2, ..., f_2^n)$, where E_n outputs the posterior probability of x belonging to all classes: $P_{E_n}^j, j \in L2$.
5. If arg$max_{j \in L2} P_{E_n}^j \in (L2 - L1)$, x can be classified by the value of arg$max_{j \in (L2-L1)} P_{E_n}^j$. The algorithm ends.
6. If arg$max_{j \in L2} P_{E_n}^j \in L1$, modify the outputs of E_n by setting $P_{E_n}^j = 0, j \in (L2 - L1)$ and $P_{E_n}^j = \frac{P_{En}^j}{\sum_{j \in L1} P_{En}^j}, j \in L1$, then take the classifier combining rule M to combine classifiers $\{f_1^1, f_1^2, ..., f_1^m, E_n\}$ and get the combined classifier E. E outputs the posterior probability of x belonging to all classes:

$P_E^j, j \in L1$.

7. Classify the test input x by the value of arg$max_{j \in L1} P_E^j$.
8. The algorithm ends.

3 Posterior Probabilistic Outputs for SVMs

Standard SVMs do not provide posterior probability information, for the reason of Vapnik's principle of never solving a problem that is more general than you actually need to solve. However, posterior probability $P(class|input)$ is very useful in practical recognition situations, such as combining several classifier's outputs for overall decision. Platt has proposed a model of SVM+sigmoid that can yield posterior probability while still retaining the sparseness of SVMs [20]. He employed a sigmoid function to approximate $P(y = 1|x)$.

$$P(y = 1|x) = \frac{1}{1 + exp(Af(x) + B)} \tag{4}$$

where, x is a test sample, $f(x)$ means the decision value of the trained SVMs, and A and B are parameters estimated by minimizing a negative log-likelihood function $l(A, B)$.

$$l(A, B) = -\sum_{i=1}^{N} t_i log(p_i) + (1 - t_i) log(1 - p_i) \tag{5}$$

where, $t_i = \frac{y_i+1}{2}$, $p_i = \frac{1}{1+exp(Af_i+B)}$, and f_i is estimated by cross-validation. Wu *et al.* improved the implementation of Platt's model and proposed a method to estimate for multi-class posterior probability by pairwise coupling [21].

4 Experiments

4.1 Datasets

In this paper, the experiment platform is PCs with 1G RAM and 3G CPU. The training algorithm is libSVM [22] with cache of 40M and kernel function of RBF.

In order to evaluate the performance of ILbyCC algorithm, experiments are run on four data sets. The first three data sets, Optical Digits Database, Vehicle Silhouette Database, and Concentric Circle Database, are took from Poliker's paper [4]. The fourth data set is a part of Yomiuri News Corpus database. The statistics of the tasks are illustrated in Table.1. The parameters used in SVMs are selected by cross-validation.

In order to test whether ILbyCC can get new knowledge from new training data while retaining the knowledge learned before in example-incremental learning scenario. Vehicle Silhouette Database was divided into three training subsets to form an incremental sequence, S_1, S_2, and S_3, each of which contains 210 instances, and a validation dataset, $TEST$, of 216 instance, near uniformly in four

Table 1. The problem statistics and the parameters used in SVMs

Data set	#attributes	#training data	#test data	#class	C	γ
Optical Digits	1024	1200	4420	10	128	0.002
Vehicle Silhouette	18	630	216	4	1500	0.00001
Concentric Circle	2	1200	500	5	128	0.125
Yomiuri News Corpus	5000	424310	87268	9	64	0.125

classes. In Optical Digits Database, 1200 instances were used as training data and all the remaining data were used for validation. The training data set was divided into six subsets to form an incremental sequence, S_1 through S_6, each of which contains 200 instances uniformly distributed over all the ten classes.

In order to evaluate whether ILbyCC can get new knowledge from new training data that correspond to previously unseen classes while retaining the knowledge learned before in class-incremental learning scenario. Concentric Circles Database was generated for testing ILbyCC's performance on incremental learning when new classes are introduced. Six training datasets, S_1 through S_6, are randomly generated for an incremental sequence. S_1 and S_2 contain 50 instances from each of the classes 1, 2, and 3; S_3 and S_4 have 50 instances from each of the classes 1, 2, 3, and 4; S_5 and S_6 have 50 instances from each of the classes 1 through 5. A validation dataset $TEST$ is randomly generated to contain 500 instances from all the five classes. From Yomiuri News Corpus database, we select all the instances of nine classes, such as crime, sport, Asian-Pacific, North-South-American, health, accident, by-time, society, and finance, which will be called as class 1 through class 9. The training data set is randomly divided into 9 incremental batches, S_1 through S_9, where S_1 through S_3 have instances from classes 1, 2, and 3; S_4 through S_6 contain instances from classes 1 through 6; and S_7 to S_9 have instances from classes 1 through 9.

In order to test ILbyCC's performance on incremental learning when different incremental step takes different parameters. Optimal parameters in each incremental step were chosen among 25 pairs of (C, γ) by 10-cross-validation. 25 pairs of (C, γ) were generated around the values of (C, γ) in Table.1 by a product factor of 2.

In order to ensure the reliability of the experimental results, the first three experiments were repeated 10 times and averaged results were presented. Only the last experiment was run one time because of its large size. In order to evaluate the performance of ILbyCC on incremental learning, several exsiting algorithms were run for a comparison study. We adopted the algorithm of Syed [5] that was denoted as ILbySV for convenience. In addition, the basic incremental learning algorithm is to use all training data gathered so far to train a classifier. In other words, when the i-th incremental batch comes, the classifiers trained before are all discarded and $S_1 \bigcup S_2 \bigcup ... \bigcup S_i$ is used to train a new classifier. For convenience, this learning way is called *Batch-training*. Obviously, Batch-training should keep all training data gotten by far, and further, catastrophic forgetting takes place when new data comes. In order to compare ILbyCC with Learn++, the paper directly quotes the experimental results of Learn++ [4]. For convenience, when all the training sessions of ILbyCC uses the same parameters, ILbyCC is denoted as ILbyCC1, when different session of ILbyCC use different parameters, ILbyCC is denoted as ILbyCC2.

4.2 Results and Analysis

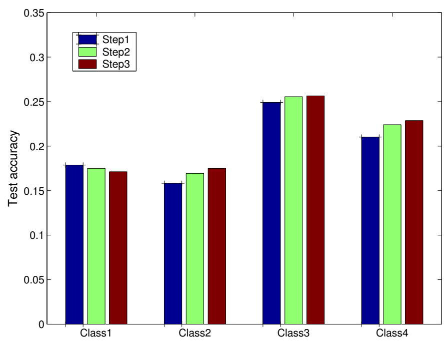

Fig. 2. The generalization performance of ILbyCC1 on each class in Vehicle Silhouette database

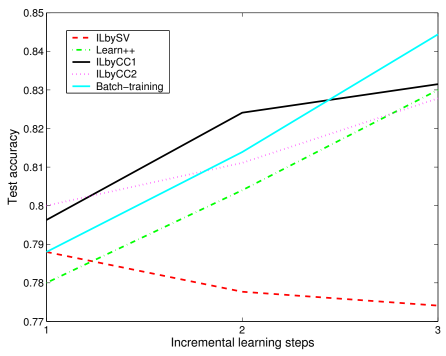

Fig. 3. Accuracy comparison of various incremental learning algorithms on Vehicle Silhouette database

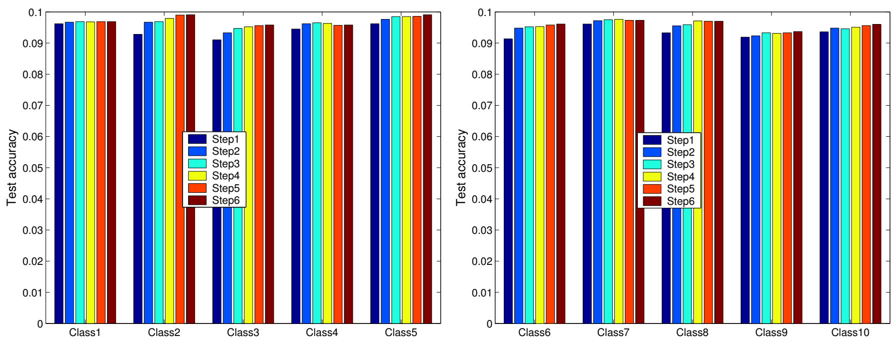

Fig. 4. The generalization performance of ILbyCC1 on each class of Optical digits database

Fig.2 presents the generalization performance of ILbyCC on each class in Vehicle Silhouette database at each incremental step. Except the test accuracy of the first class drops slightly during incremental learning procedure, the test accuracy of the rest class increase when new examples are introduced, indicating ILbyCC was able to preserve the knowledge learned before and acquire new information. As seen in Fig.3, ILbyCC can incrementally learn successfully. ILbyCC1 is slightly good then Learn++. ILbyCC1 and ILbyCC2 have nearly the

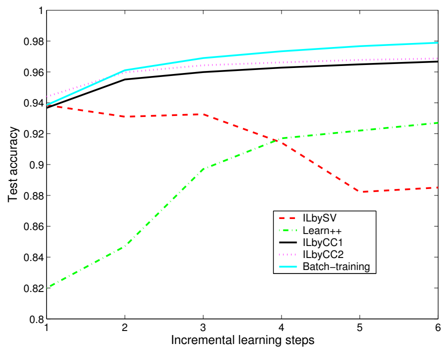

Fig. 5. Accuracy comparison of various incremental learning algorithms on Optical digits database

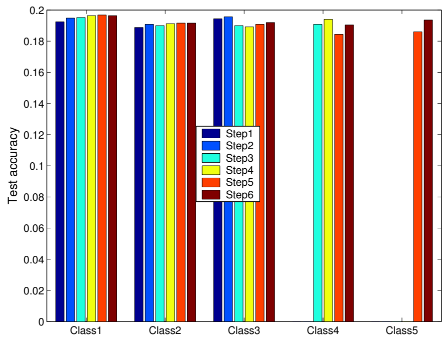

Fig. 6. The generalization performance of ILbyCC1 on each class of Concentric Circle database

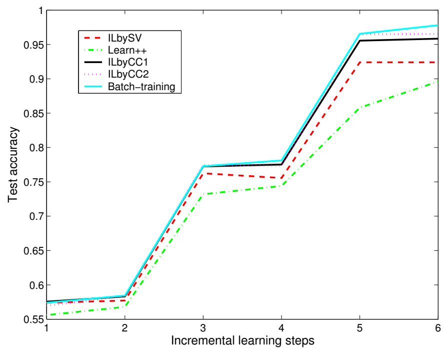

Fig. 7. Accuracy comparison of various incremental learning algorithms on Concentric Circle database

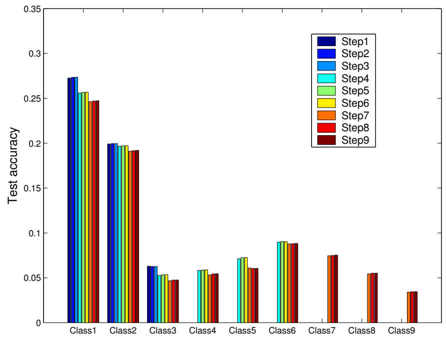

Fig. 8. The generalization performance of ILbyCC1 on each class in Yomiuri News Corpus database

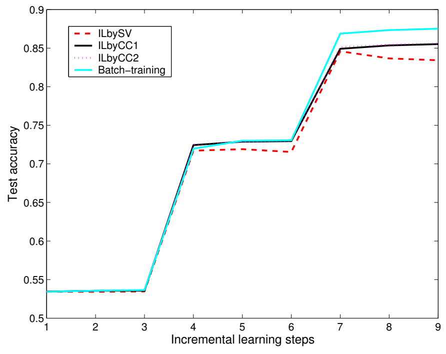

Fig. 9. Accuracy comparison of various incremental learning algorithms on Yomiuri News Corpus database

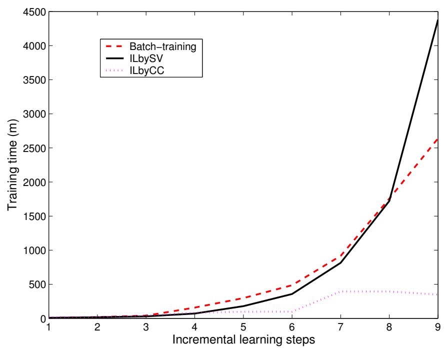

Fig. 10. Comparison of training time on Yomiuri News Corpus database

same generalization ability. Because all incremental batches are not always in the same distribution, the incremental learning performance of ILbySV fluctuates.

Fig.4 illustrates the generalization performance of ILbyCC on each class in Optical digits database at each incremental step. It can be seen that the test accuracy of ILbyCC on each class nearly uniformly increase, indicating the learned knowledge don't lost and new knowledge are acquired. Fig.5 illustrates that ILbyCC has nearly the same incremental learning performance with Batch-training. ILbyCC works better that Learn++ and ILbySV. ILbyCC2 performances well than ILbyCC1. The generalization ability of ILbySV fluctuates.

Fig.6 shows the generalization performance of ILbyCC on each class in Concentric Circle database at each incremental step. It can be seen that, during the whole incremental learning procedure, the test accuracy of ILbyCC on class-1 and class-2 increase gradually while the test accuracy of ILbyCC on the other classes first decrease slightly when new classes are introduced and increase when training data with the same class labels are continuously added, indicating that ILbyCC can preserve the learned knowledge. In Fig. 7, a large improvement on the performance is obtained after the third and fifth incremental steps, since these incremental steps introduced new classes that were not available earlier. However, only minor improvements in the performance can be observed from the test accuracy curves when new classes are not introduced. Fig.7 shows that ILbyCC1 and ILbyCC2 performance as good as Batch-training and better than Learn++ and ILbySV. ILbyCC2 works a little better than ILbyCC1.

Fig.8 presents the generalization performance of ILbyCC on each class in Yomiuri News Corpus database at each incremental step. Note that, in class-incremental learning scenery, the introduced examples with new class label can make the test accuracy of ILbyCC on the earlier classes to decrease in a small degree, for the reason of that new introduced classes make classification planes between classes more clear. In example-incremental learning scenario, the test accuracy of ILbyCC1 on each class increase. Therefore, ILbyCC can preserve the knowledge learned before even new class introduced. In Fig.9, there are larger improvements on the generalization performance when new classes are introduced, indicating that ILbyCC can learn from new introduced classes. From Fig.9, it can be noted that ILbyCC and Batch-training have nearly the same generalization ability. In Fig.10, it can be seen that the training time of ILbyCC is far smaller than the training time of Batch-training and ILbySV. The large speedup of ILbyCC can compensate the slight decrease of its generalization performance compared with Batch-training.

Why can ILbyCC work effectively? According to the theory of bias-variance [23], decomposing training data will introduce bias and makes the generalization ability of single classifier decrease, however, decomposing training data will increase the variances between all classifiers and increase the generalization ability of the combined classifier, which compensates the decrease of the generalization ability caused by decomposition. Therefore, ILbyCC has nearly the same test accuracy with Batch-training. In addition, the combining rule (2) can automatically invalidate the classifiers that is not much confident of its outputs, i.e., given $P_j(y = 1|x) \approx ... \approx P_j(y = k|x)$, the result of the equation (3) will not be influenced by the outputs of the j-th classifier. Therefore, Averaged Bayes can automatically select the classifiers that is confident of its outputs to combine.

Note that the performance of ILbyCC1 and ILbyCC2 in all the simulations are nearly the same, it is very interesting to observe that the time complexity for selecting optimal parameters is decreased by training data decomposition. It is not reasonable for incremental learning algorithm to wait for all training data to select optimal parameters. It is also not reasonable to apply the parameters, which is gotten from the first incremental batch, to the following incremental steps. Therefore, ILbyCC not only decreases the time complexity of parameter selection but also makes incremental learning possible.

4.3 Discussions

Compared with Learn++, the proposed ILbyCC satisfies the criteria proposed by Polikar and has comparable incremental learning ability, but ILbyCC can be implemented more simply. Learn++ is a kind of AdaBoost in essence, Learn++ should use more parameters and train more classifiers. Note that ILbyCC is a bagging-like algorithm, ILbyCC can be parallized for training speedup, while Learn++ can only be implemented in serial. In addition, ILbyCC needs no communication between classifiers, it can well protect the privacy of data. ILbyCC is a good application of SVMs that can outputs posterior probabilistic, the work in this paper proved the availability of the algorithm estimating the posterior probabilistic of SVMs. From the point of author, this paper is the first application to apply posterior probabilistic SVMs to real problem.

5 Conclusions

In this paper, we have proposed a novel incremental learning algorithm, ILbyCC that uses Averaged Bayes rule to combine classifiers. The experimental results indicate that ILbyCC can not only preserve the knowledge learned before but also can learn new knowledge from new added data and further new knowledge from new introduced classes. Three main advantages of ILbyCC over existing algorithms are simply implementing, small time complexity for parameter selection, and training time saving. In addition, the proposed algorithm is a general framework of incremental learning and any machine learning algorithm that can output posterior probabilistic can be integrated into ILbyCC.

References

1. Jantke, P.: Types of Incremental Learning. AAAI Symposium on Training Issues in Incremental Learning, March 23-25, Standford CA, 1993
2. Zhou, Z.H. and Chen, Z.Q.: Hybrid Decisions Tree. Knowledge-Based System, 15 (2002) 515-528

3. Syed, N.A., Huan, L., and Sung, K.K.: Handling Concept Drifts in Incremental Learning with Support Vector Machines. In: Proceedings of KDD-99, San Diego, CA,USA, 1999
4. Polikar, R., Udpa, L., Udpa, S.S., and Honavar, V.: Learn++: An Incremental Learning Algorithm for Supervised Neural Networks, IEEE Transaction on Systems, Man, and Cybernetics, 31 (2001) 497-508
5. Syed, N.A., Liu, H., and Suang, K.K.: Incremental Learning with Support Vector Machines, In: Proceedings of IJCAI-99, San Diego, CA, USA (1999)
6. Domeniconi, C. and Gunopulos, D.: Incremental Support Vector Machine Construction. In: Proceedings of the 2001 IEEE International Conference on Data Mining, San Jose, California, USA, (2001) 589-592
7. Cauwenberghs, G. and Poggio, T.: Incremental and Decremental Support Vector Machine Learning. In: Advances in Neural Information Processing Systems, MIT Press, 13 (2001) 409-415.
8. Diehl, C.P. and Cauwenberghs, G.: SVM Incremetal Learning, Adaptation and Optimization. In: Proceedings of IJCNN-03 (2003) 2685-2690
9. Ralaivola, L. and d'Alch*é*-Buc, F.: Incremental Support Vector Machine Learning: a Local Approah. Lecture Notes in Computer Science, 2130 (2001) 322-329
10. Liu, Y.G., He, Q.M., and Chen, Q.: Incremental Batch Learning with Support Vector Machines. In: Proceedings of the 5th World Congress on Intelligence Control and Automation, Hangzhou, P.R. China (2004) 1857-1861
11. Grossberg, S.: Nonlinear Neural Networks: Principles, Mechanisms and Architectures. Neural Networks, 1 (1988) 17-61
12. Fu, L.M., Hsu, H.H., and Principe, J.C.: Incremental Back-propagation Learning Networks. IEEE Transaction on Neural Networks, 7 (1996) 757-761
13. R*ü*ping, S.: Incremental Learning with Support Vector Machines. In: Proceedings of the IEEE International Conference on Data Mining, San Jose, CA (2001)
14. Lu, B.L., and Ito, M.: Task Decomposition and Module Combination Based on Class Relations: a Modular Neural Networks for Pattern Classification. IEEE Transaction on Neural Networks, 10 (1999) 1244-1256
15. Zhou, Z.H. and Chen S.F.: Neural Network Ensemble. Chinese J.Computers (in Chinese), 25 (2002) 1-8
16. Lu, B.L. and Ichikawa, M.: Emergent Online Learning in Min-max Modular Neural Networks. In: Proceedings of IJCNN'01 (2001) 2650-2655
17. Macek, J.: Incremental Learning of Ensemble Classifiers on ECG data. In: Proceedings of CBMS'05 (2005)
18. Wang, H.X., Fan, W., Yu, P.S., and Han, J.W.: Mining Concept-drifting Data Streams Using Ensemble Classifiers. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (2003)
19. Xu, L., Krzy*ż*ak, A., and Suen, C.Y.: Methods of Combining Multiple Classifiers and Their Application to Handwriting Recognition. IEEE Transaction on Systems, Man, and Cybernetics, 22 (1992) 418-434
20. Platt, J.C.: Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In: Advances in Large Margin Classifiers, MIT Press (1999)
21. Wu, T.F., Lin, C.J., and Weng, R.C.: Probability Estimates for Multi-class Classification by Pairwise Coupling. Journal of Machine Learning Research, 5 (2004) 975-1005
22. Chang, C.C. and Lin, C.J.: LIBSVM-A Library for Support Vector Machines. [online] Available: *http://www.csie.ntu.edu.tw/∼cjlin/libsvm*
23. Breiman, L.: Bagging Predictors. Machine Learning, 24 (1996) 123-140