Adaptive Incremental Learning with an Ensemble of Support Vector Machines

Marcelo N. Kapp, Robert Sabourin
École de technologie supérieure
{*kapp,rsabourin*}*@livia.etsmtl.ca*

Patrick Maupin
DRDC Valcartier - Canada
patrick.maupin@drdc-rddc.gc.ca

Abstract

The incremental updating of classifiers implies that their internal parameter values can vary according to incoming data. As a result, in order to achieve high performance, incremental learner systems should not only consider the integration of knowledge from new data, but also maintain an optimum set of parameters. In this paper, we propose an approach for performing incremental learning in an adaptive fashion with an ensemble of support vector machines. The key idea is to track, evolve, and combine optimum hypotheses over time, based on dynamic optimization processes and ensemble selection. From experimental results, we demonstrate that the proposed strategy is promising, since it outperforms a single classifier variant of the proposed approach and other classification methods often used for incremental learning.

1. Introduction

A problem when updating classification systems is that their best set of parameters can change as a result of data coming in at different times. These changes would constitute minor fluctuations in the underlying probability distributions, which could result from either sample shifting or the natural evolution of classification problems. As a consequence, the sample distributions of training data chunks may change during incremental learning processes, affecting the system in several ways. In the literature, this phenomenon is defined as *population drifts* [4, 6]. Thus, unless an incremental learning system considers readjusting its internal parameters in relation to data variations beyond merely training its existing models with new data, the whole system may become obsolete, and hence fail to achieve a better adaptation in the future. In light of this, unlike common incremental learning methods that consider the adjustment of parameters as a static process (i.e. constant parameter values are employed infinitely), we propose to optimize them over time to increase the system's power of generalization and decrease its complexity. The main novelty of this paper is to consider the incremental learning process as a dynamic optimization one, in which multiple hypotheses are dynamically tracked, evolved, and combined over time. The proposed approach incorporates various techniques, such as incremental Support Vector Machine (ISVM) classifiers, change detection, dynamic Particle Swarm Optimization (DPSO), and, finally, dynamic selection of classifier ensembles (EoC). Basically, it is based on these two principles: (1) incremental accommodation of new data by updating models, and (2) dynamic tracking of new optimum system parameters for self-adaptation. Incremental SVM ensembles are employed for two main reasons: (1) the SVM classifier does not depend on the dimensions of the input space, which makes it robust with respect to the well known *curse of dimensionality* and very advantageous for incremental learning situations; and (2) SVMs are combined into ensembles because their performance can often surpass that of a single classifier, especially when the level of uncertainty is high, i.e. when only small sample sets are available for training [7]. The proposed framework will also ultimately contribute to strategies for optimizing and overproducing classifiers, as well as to the application of memory-based mechanisms for solving dynamic optimization processes. We test the proposed approach in single and multiple classifier configurations and compare them with these strategies: SVM optimized with PSO in batch mode, incremental SVM with parameter values fixed beforehand, and two incrementally capable classifiers (1-NN and Naïve Bayes). We also try to verify whether or not incremental learning with SVM can achieve similar performances to those obtained in batch mode, and whether or not the adaptation of the system's parameters over time is actually a dynamic optimization problem, and hence important to achieving high performances.

2 Proposed Approach

From the literature, it can be noted that, no matter what the incremental learning approach, no consideration has been given to the tuning of system parameters over time. In other words, systems are always updated based on the same fixed parameter values, or at the classifier combination levels in ensemble approaches. Thus, the updating of classifiers with the adaptation of their parameters has not yet been investigated. Furthermore, recent results indicate that using well-tuned incremental ensemble learners could achieve better than merely moderate performances [5]. In this connection, we propose an approach for adaptive incremental learning that regards the incremental learning process as a dynamic optimization one. In particular, it performs adaptive incremental learning by optimizing, selecting, and combining incremental SVM classifiers over time. More specifically, it is designed to dynamically indicate optimum solutions for sequences of datasets $\mathcal{D}(k)$, either by using the best solutions found to that point, or by starting new dynamic optimization processes. As we employ ISVMs as our base classifiers and DPSO for searching for optimum hyperparameter values, each solution $\mathbf{s}$ represents a particle codifying an SVM hyperparameter set, e.g. $\{C, \gamma\}$. Change detection mechanisms monitor novelties in the objective function $\mathcal{F}$, and indicate how the system must act. The models generated are updated from incoming data, and then dynamically selected and combined into an ensemble $\mathcal{C}^*$. The method is implemented based on a framework composed of five main modules: change detection, adapted grid-search, DPSO, incremental SVMs, and decision fusion. It represents some upgrades implemented since our first version was introduced in [3], such as the use of incremental classifiers, dynamic selection, and the building of ensembles from optimized models. The way in which the proposed approach works is set out in Algorithm 1. Below, we detail each module. Δ represents a set of data sv composed of support vectors and relevant samples rs selected during the training of the final classifier from the best particle $\mathbf{s}^*$. Therefore, $\Delta = \{sv^* \cup rs\}$, where sv^* refers to support vectors obtained from the final incremental model $\mathcal{M}^*$ trained with hyperparameters found by the best particle $\mathbf{s}^*$. SV denotes the set of support vectors sv from incremental models obtained after final training of all P particles from a Swarm $\mathcal{S}(k-1)$, i.e. $SV = \{sv_j\}_{j=1}^{P}$. $\mathcal{C}$ represents an ensemble composed of all models (i.e. classifiers) $\mathcal{M}_i$. So, $\mathcal{C} = \{\mathcal{M}_i\}_{i=1}^{P}$, where P is the maximum number of optimized solutions. Finally, for the sake of simplicity, in the equations, $\mathcal{D}(k)$ represents the merging of new data and the current knowledge stored.

Algorithm 1 Adaptive Incremental Learning (AIL)

Input: A training set of data $\mathcal{D}(k)$.
Output: Optimized SVM classifier/ensemble.
recall_system_memory_stm($\mathbf{s}^*(k-1)$,$\mathcal{S}(k-1)$)
if there is a $\mathcal{S}(k-1)$ **then**
  Check the preceding best solution $\mathbf{s}^*(k-1)$ regarding the dataset $\mathcal{D}(k)$
  if Change_Detection($\mathbf{s}^*(k-1)$,$\mathcal{D}(k)$) **then**
    Activate the adapted grid-search module and get solution $\mathbf{s}'(k)$
    if Change_Detection($\mathbf{s}'(k)$,$\mathcal{D}(k)$) **then**
      Activate the DPSO module
    end if
  end if
else
  Activate the DPSO module
end if
upgrade_system_memory_stm($\mathbf{s}^*(\cdot)$,$\mathcal{S}(\cdot)$).
Train/update/combine the final incremental SVM classifiers from incoming data $\mathcal{D}(k)$, $\Delta(k)$, and SV.

• **Change Detection:** This module monitors differences in the objective function values, in this case cross-validation error estimations ϵ obtained for a best solution $\mathbf{s}^*$ on the datasets $\mathcal{D}(k-1)$ and $\mathcal{D}(k)$, i.e. $\epsilon(\mathbf{s}^*, \mathcal{D}(k-1))$ and $\epsilon(\mathbf{s}^*, \mathcal{D}(k))$, respectively. If the solution is found not to be satisfactory for the process, then a further searching level is activated. The adequacy of a solution is related to a stable region, i.e. if the objective function value computed does not lie in a "stable" region, which is computed through the maximum expected difference δ_{max} between the objective function values at the 90% confidence level using a normal approximation to the binomial distribution (see Equations 1 and 3) [2].

$$\delta_{max} = z_{0.9} \times \sqrt{\sigma} = 1.282 \times \sqrt{\sigma} \tag{1}$$

Where σ is computed by, where $W(\cdot)$ is the dataset size:

$$\sigma = \frac{\epsilon(\mathbf{s}^*, \mathcal{D}(k-1)) \times (1 - \epsilon(\mathbf{s}^*, \mathcal{D}(k-1)))}{W(\mathcal{D}(k-1))} + \frac{\epsilon(\mathbf{s}^*, \mathcal{D}(k)) \times (1 - \epsilon(\mathbf{s}^*, \mathcal{D}(k)))}{W(\mathcal{D}(k))} \tag{2}$$

• **Adapted Grid-Search:** This module identifies optimum solutions through the re-evaluation of knowledge acquired from previous optimizations carried out by our DPSO module with respect to the current data $\mathcal{D}(k)$. This knowledge is represented by a set $\mathcal{S}(k-1)$ of optimized solutions stored in a short term memory (STM).

• **Dynamic Particle Swarm Optimization - DPSO:** The DPSO module indicates new optimum solutions by means of dynamic optimization processes, the goals of which are: (1) to work on multi-modal search spaces,

and (2) to track changes of fitness landscapes and optimum point positions, since these can vary depending on the incoming data received over time. It is based on the PSO algorithm combined with dynamic optimization techniques. We implement this module so that it is capable of exploring multiple regions in parallel, and is therefore a better fit for functions with possible *multiple local optima*. A full description of this module and detailed explanations about each of the previous modules can be found in [3].

• **Incremental Support Vector Machine - ISVM:** In this study, we implement an incremental SVM version based on the Syed et al. method [6]. As in [6], an incremental SVM model $\mathcal{M}_i(k)$ is trained on the current training data chunk $\mathcal{D}(k)$ and its historical support vectors $sv(k-1)$ identified from a previous learning event at a given time k. However, unlike in [6], where only support vectors are stored, our incremental SVM module also retains additional training samples relying on a "relevant region" which exceeds the SVM margins in half of their sizes. Although the storage of additional samples is not a desirable property in incremental learning algorithms, it is necessary, because these additional samples can become support vectors during optimizations of SVM hyperparameters optimization in the future.

• **Decision Fusion Module:** Our dynamic selection strategy is implemented based on a generalization bound introduced in [1]. In this dynamic strategy, only combinations of classifiers that minimize this bound (called the CI measure here) are selected to make up the final ensemble. In particular, this measure is computed as $CI = \sigma(\tau)/\mu(\tau)^2$, where σ and μ denote the variance and the average calculated over the set of margins τ from samples of the current training set respectively. The margin of a sample $\mathbf{x}_i$ represents a degree of confidence in its classification. Basically, it is calculated as the difference between the decision support ϑ assigned to the true class t minus the highest support estimated for any other class j, i.e. $\tau_i = \vartheta_t(\mathbf{x}_i) - \max_{\substack{j=1,\ldots,c \\ j\neq t}}\{\vartheta_j(\mathbf{x}_i)\}$. Here, for a single classifier, the decision support for a class j is denoted as the posterior probability assigned to it. In the same way, for an ensemble composed of classifiers with output probabilities, the decision support for a class j is the average over the posterior probabilities assigned to it by each member. The selection process is performed as follows. First, the pool of classifiers $\mathcal{C}(k)$ generated from $\mathcal{S}(k)$ is sorted according to each classifier's individual confidence level (average margins). Then, the selection process starts by adding one classifier at a time until the maximum number of classifiers is reached, i.e. a number of particles P. Every time a classifier is added, the CI selection criterion is recomputed. The best ensemble selected $\mathcal{C}^*$ is that with a minimal CI value. Thus, the key idea is to select the ensemble with the strongest, i.e. highest, confidences, and fewer correlated classifiers over the current training set. Finally, once the best ensemble $\mathcal{C}^*$ has been selected, the classifiers are combined using weighted average voting based on their performances.

3. Experimental Protocol

The following experimental protocol has been carried out to test our adaptive incremental learning approach. First, to characterize the occurrence of population drifts with greater impact, the original training sets were divided into small datasets. The total number of chunks and their sizes were determined based on a minimum number of samples required for each class, which was set to at least 16. Database descriptions and number of chunks used are listed in Table 1. The results represent averages drawn over 10 replications.

Table 1. Databases used.

Databases	# of Classes	# of Features	# of chunks	Training Size	Test Size
DNA	3	180	29	2,000	1,186
IR-Ship	8	11	8	1,785	760
P2	2	2	120	3,856	10,000
Satimage	6	36	25	4435	2,000

The following strategies were used for comparison purposes: Batch SVM-PSO, the original training datasets are used for selecting optimum SVM hyperparameters with PSO and generating the final model; Incremental *no-less* classifiers (1-Nearest Neighbor (1-NN) and Naive Bayes (NB)); Incremental SVM (ISVM), where an incremental SVM classifier tailored from [6] is updated from successive data chunks $\mathcal{D}(k)$ (its hyperparameters are first tuned with PSO over the first data chunk $\mathcal{D}(1)$, and then kept fixed over all the other data chunks); our proposed incremental approach in single classifier mode (IS-AIL), ; and Incremental EoC-AIL (IEoC-AIL), our proposed approach in EoC mode. We have used these parameter settings for the optimization algorithm. The swarm sizes were set to 20. The dimensions of the (C and γ) search space were set to $[2^{-6}, 2^{14}]$, $[2^{-15}, 2^{10}]$.

3.1 Results

We report the generalization errors achieved by each strategy tested in Table 2. The best results are shown in bold. The underlined values indicate when one incremental strategy was significantly better than the others,

according to a Kruskal-Wallis nonparametric statistical test. By analyzing the results in this table, we can see that the SVM is very promising for incremental learning, even with its hyperparameters kept fixed at a value found on $\mathcal{D}(1)$ (ISVM). More importantly, we observe the efficiency of the proposed method, as well as conclude that adaptive incremental learning clearly leads to better performances. That is because the single classifier version of our proposed method (IS-AIL) achieved better results than the ISVM strategy commonly used. This shows the importance of the adaptation of hyperparameters and the use of relevant samples during the incremental learning process. In addition, we can see that the proposed method (IEoC-AIL) achieved results similar to those of SVM-PSO in batch mode, sometimes even better, proving that the dynamic selection and combination of optimum solutions can actually improve the overall performance of the system.

Table 2. Mean and standard deviation of error rates obtained.

Approaches tested	Databases			
	DNA	IR-Ship	P2	Satimage
SVM-PSO	5.13 (0.18)	4.86 (0.35)	1.64 (0.10)	8.06 (0.13)
1-NN	23.69	9.21	**2.49**	10.95
NB	6.32	30.92	42.38	20.45
ISVM	8.43 (1.48)	7.93 (0.44)	5.24 (0.14)	22.15 (0.93)
IS-AIL	4.71 (0.25)	5.04 (0.55)	4.80 (0.90)	8.83 (0.27)
IEoC-AIL	**4.61 (0.27)**	**4.03 (0.30)**	3.17 (0.56)	**8.14 (0.17)**

We also show results involving the generalization errors between the IS-AIL and IEoC-AIL strategies over a replication for the Satimage database in Figure 1. It can be seen that different cardinalities are obtained throughout the process. Eventually, the classical "non-less" incremental learners NB and 1-NN achieved the worst performances. The only exception was the P2 database, where the 1-NN classifier outperformed the other methods tested. We note that the dynamic adaptation of the hyperparameters during the incremental learning process (IS-AIL) seemed to converge to the results obtained in batch mode (SVM-PSO). For example, it also tended to identify about the same number of support vectors as when all the data are available for training.

4. Conclusion

We have proposed a modular dynamic optimization approach to perform adaptive incremental learning, which generates classifiers from optimum regions of the parameter search space and then dynamically selects ensembles based on the classifiers' confidence levels to improve the overall results. We have empirically demonstrated that the dynamic optimization of an incremental classification system could improve its performances, so that they could overcome classifiers without adaptation or other classical methods. Moreover, we have shown that the use of a multiple classifier approach makes the system more flexible and robust in performing incremental learning.

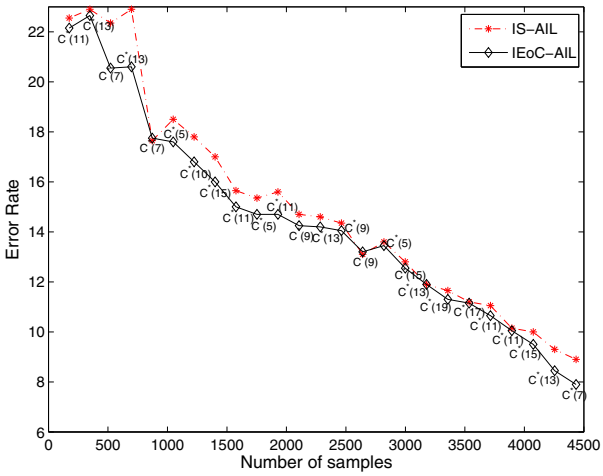

Figure 1. Results for IS-AIL and IEoC-AIL.

5 Acknowledgments

This research was supported by grant OGP0106456 to Robert Sabourin from NSERC of Canada.

References

[1] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[2] L. Cohen, G. Bakish, M. Last, A. Kandel, and O. Kipersztok. Real-time data mining of non-stationary data streams from sensor networks. *Inf. Fusion*, 9(3):344–353, 2008.

[3] M. N. Kapp, R. Sabourin, and P. Maupin. A PSO-based framework for dynamic svm model selection. In *Procs of the GECCO*, pages 1227–1234, 2009.

[4] M. G. Kelly, D. J. Hand, and N. M. Adams. The impact of changing populations on classifier performance. In *Procs of 5th Int. Conf. on KDD*, pages 367–371, 1999.

[5] D. Parikh and R. Polikar. An ensemble-based incremental learning approach to data fusion. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(2):437–450, April 2007.

[6] N. A. Syed, H. Liu, and K. K. Sung. Handling concept drifts in incremental learning with support vector machines. In *Procs of the 5th KDD*, pages 317–321, 1999.

[7] G. Valentini. An experimental bias-variance analysis of SVM ensembles based on resampling techniques. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 35(6):1252–1271, 2005.