SBL-PM-E and SBL-PM-M: New Methods for Partial Memory Learning

Karol Grudziński

Department of Physics, Academy of Bydgoszcz
Plac Weyssenhoffa 11, 85-072 Bydgoszcz, Poland
E-mail: kagru@ab-byd.edu.pl

Abstract. Partial Memory Learning (PML) is a machine learning paradigm in which only subset of a training set is used during learning. This paper concerns new methods for partial memory learning. The SBL-PM-E method is an extension of the SBL-PM algorithm developed by us earlier. The SBL-PM-M method is however a completely new model. We evaluate the performance of the new algorithms on several real-world datasets and compare them to the original SBL-PM algorithm.

keywords: classification, machine learning, partial-memory learning

1 Introduction.

Partial Memory Learners (PML) are usually on-line learning systems that select and store portion of the past learning examples. This paper concerns new methods for partial memory learning.

The first of the proposed algorithms, SBL-PM-E (Enhanced), extends our SBL-PM-B (formerly called SBL-PM) system developed earlier []. SBL-PM-B (B stands for Basic) is a batch-learning system which aim is to reduce training set size for better understanding of data through case-based data explanation. The second aim of constructing SBL-PM-B algorithm was to generate data for other computational intelligence (CI) models to allow for much faster learning (only small portion of the training data is retained usually at the expense of slightly lower classification accuracy on unseen cases). The SBL-PM-E has been constructed to avoid weakneasses of the SBL-PM-B and to allow for increase of the prediction ability at no additional computational cost. SBL-PM-M finds an optimal training set and is a completely new method based of minimization of the cost function, which is the number of errors the classifier makes.

The paper consists of ... sections. First the old SBL-PM-B algorithm is shortly described. Next the weaknesses of this model are enumerated and natural ways to extend this method are summarized which lead to the presentation of the SBL-PM-E algorithm, which contains enhancements described earlier implemented in software. In the third section a new partial-memory algorithm which we call SBL-PM-M is described. In the fourth section the generalization abilities of all classifiers are compared on real world datasets. The last section concludes the paper.

2 The SBL-PM-B System

SBL-PM-B (formerly SBL-PM) is our own algorithm that has already been described earlier []. The name SBL-PM-B is slightly misleading as it imposes connection of this algorithm with Similarity Based Methods. SBL is the name of our package, Similarity Based Learner, in which this algorithm had been first implemented. PM stands for Partial Memory and B for Basic. This algorithm is however universal and can be built on top of arbitrary classifier. A description of SBL-PM-B follows.

Cases are removed from the training set taking into account the leave-one-out classification accuracy value on the training set and the value computed by classifying training set as the test set, given partial memory as training set. A quotation from our earlier paper [] provides the best description of the SBL-PM-B algorithm:

1. 'Set the partial memory of the system (reference set) to the entire training set, $\mathcal{R}$=$\mathcal{T} = R_i, i = 1..N$.
2. Set the target accuracy Δ performing the leave-one-out test on $\mathcal{T}$; lowering the target accuracy will reduce the final number of the reference vectors.
3. For i=1 to N
   (a) Set the temporary reference set to $\mathcal{R}$'=$\mathcal{R}-R_i$.
   (b) Using the leave-one-out test and the current reference set $\mathcal{R}$' calculate the prediction accuracy A_c on the whole training set $\mathcal{T}$.
   (c) If $A_c \geq \Delta$ set $\mathcal{R}$=$\mathcal{R}$'

Vectors are sequentially eliminated from the reference set if the classification accuracy drops below the accuracy Δ obtained in the leave-one-out test on the entire training set, the case from the reference set must be retained, otherwise it can be eliminated. The threshold value $(0 < \Delta < 100)$ (in percents) may also be given by the user, allowing for some degradation of the performance as a penalty for reduction of the reference set. The final reference set should be significantly smaller than the original training set with minimal degradation of the prediction accuracy. By setting the value of Δ at the begining of calculations the number of cases that will remain in the partial memory is controled to a certain extent. Δ can be optimized running the SBL-PM procedure for several values of Δ'.

3 The SBL-PM-E Algorithm

One of the significant weaknesses of SBL-PM-B is no control on the distribution of instances remianing in partial memory. Optimization of the Δ parameter permits only to control the amount of cases left in partial memory. Our exepriments indicate that often there are no instances retained in partial memory for one or more classes after the learning phase. This significantly degrades the performance of a classifier. SBL-PM-E takes care about preserving a proper distribution of samples.

The second enhancement introduces additional way to reduce the amount of the instances in partial memory. Once the first run is finshed the system proceeds

starting from the partial memory generated in a previous step. The number of runs can be controlled by a parameter that is set before calculation is started. Additional parameter which forces to retain given in advance number of samples in partial memory has also been introduced to SBL-PM-E.

Another issue is a prediction ability of the system. SBL-PM-B had been designed to minimize the partial memory size mainly in order to provide case-based data analysis as an alternative to rule based data explanation. With starting to implement successors of SBL-PM-B, such as the enhanced version SBL-PM-E, we are more interested in achieaving as good generalization as possible with minimal partial memory size. In SBL-PM-E with each case removal the accuracy on the training set is noted and the final partial memory set is this one with optimal training accuracy. Additionally two windows are provided to which this optimization is restricted. First window is constructed by specifying two indeces of samples from a training set. This permits to restric optimization to, say last 50 cases in partial memory. The second window is constructed by giving upper and bottom training accuracy boundaries. It should be noted that without this restriction, if the first window covers the entire training set, preference for optimal partial memory is given to 100% training accuracy usually obtained with partial memory equal to the entire training set.

The detailed algorithm description of SBL-PM-E will not be presented due to complexity of this method and many nested conditional expressions.

4 The SBL-PM-M Algorithm

The idea of SBL-PM-M is very simple. Each training vector is assigned a binary weight with the value of 1 indicating that this case takes part in the classification process and otherwise is not taken into account. The number of weights is equal to the number of samples in the training set which are optimized with non-gradient minimization routine. The cost function is the number of errors the classifier makes. This process corresponds to selection of attributes through minimization (also possible with the SBL [] software) and does not lead to overfitting as the number of cases is usually a small portion of the entire training set. So far we have tried only the simplex minimization method [].

Such a minimization process is much harder than in the case of weighting of attributes since the number of adaptive parameters in reference selection is usually much higher than in the case of weighting of attributes. However proper construction of the minimization termination criterion allows to arrive at resonable solution already after 150 – 200 cost evaluations. The total number of cost evaluations is increased by the number equal to the number of training samples + 1 which is required to initialize the simplex.

One could try, instead of binary weights, to employ real parameters and weight the importance of each reference case in nearest neighbor classification. The number of adaptive parameters starts to be high but one might use regularization techniques to eliminate redundant reference cases. Besides, the number of weights in large MLP networks is higher than the number of cases so overfitting should not be a

problem. Also one could terminate the convergence of the method earlier using not completely trained model and thus avoiding overfitting.

5 Stabilization of the results

Stochastic PML systems such as SBL-PM-M as well as the other described in this paper PML algorithms suffer from a relatively high instability. Instability of a model is defined as sensitivity of the training and test accuracies on data perturbation. Low variance of classification accuracy characterizes stable models. In order to stabilize the results, committees (ensembles) of models are used. The most common are majority committees where calculations are repeated several times and the predicted class with the highest probability is selected.

5.1 Prototype-based Committees

Most of computational intelligence systems are trained on the entire training partitions. However in PML systems described here, either randomization of a training partition or stochastic nature of SBL-PM-M, leads to different cases that are retained in partial memory. To stabilize the classification and to provide better, more reliable case-based data analysis, introduction of prototype-based committees is proposed.

The idea is very simple: the model is run several times and for each training instance the occurrence of a given case in partial memory is noted. A level of competence should be introduced, being a threshold reaching of which indicates that a given instance should be placed in partial memory of a target system. To illustrate it better consider three cases, c_1, c_2, c_3, belonging to the original training set $\mathcal{T}$. Assume that the partial memory construction has been repeated 10 times and the level of competence $L_c = 5$. Assume also that case c_1 has been retained in partial memory eight times, c_2: six times and c_3 three times. With a given level of competence of five, cases c_1 and c_2 will be retained in partial memory and the sample c_3 will be rejected. However setting L_c to 3 will make all the cases be retained. The level of competence is a way of controlling the size of the partial memory and the extent to which the classification results should be stabilized. This parameter may be optimized to reach the compromise between the level of stability and classification accuracy. Sufficiently large number of cases in partial memory should be retained in each run in order to increase the probability of occurrence of a given sample in a reference set.

The research on prototype-based committees is quite advanced and the results will be published in a subsequent paper.

6 Numerical experiments

6.1 Learning and Evaluation of Models

In order to select the optimal model from a pool of methods, accuracies on training or validation sets should be compared and the best model is the one with the highest

accuracy. All models belonging to the SBM framework, with exception of PML methods, use cross-validation result on a training partition to estimate their training accuracy. The best model is then selected and evaluated on unseen cases.

By $\mathcal{R}$we will denote a reference (partial memory) and by $\mathcal{T}$a training set. In order to estimate their training accuracies PML methods require the test to be performed on the $\mathcal{T}$- $\mathcal{R}$set with the $\mathcal{R}$set as a training set. Consider partial memory set $\mathcal{R}$consisting of the number of cases equal to the number of classes in a problem domain. If all the cases from $\mathcal{R}$belong to distinct classes, cross- validation of $\mathcal{R}$will always lead to 0% value of training accuracy, indicating that this model will perform very poorly on unseen data, which may not be true. Besides, if the number of classes is low, (ex. 3), only maximum of 3-fold cross- validation can be performed whilst other (non-PML) models are usually trained with 10-fold stratified cross-validation or leave-one-out test. It should be stressed here that SBL software in its most recent version permits to select whether learning should be conducted through cross-validation or by performing test on the $\mathcal{T}$- $\mathcal{R}$set taking $\mathcal{R}$as a training set. However the final training accuracy reported is always computed by performing a test on a reduced training set with partial memory as the training set. We tried both learning strategies in our numerical experiments with leave-one-out learning as a cross-validation test. It should be noted that, taking into account the way the kernel of the SBL program is constructed, leave-one-out test gives always the same result and the accuracy does not depend on the randomization of a training partition. Performing cross- validation with lower folds requires computing average of results of several iterations but it still confuses the minimization subroutine as a model with given adaptive parameters gives different values of the cost function with differently randomized training partitions. This makes minimization not converge both in attribute as well as case weighting.

Comparing how well PML and non-PML models are trained requires further investigation. PML methods have not been used in SBM metaleraning[] mainly because of inability to select the best model from among PML and non-PML methods. In the case of plain nearest neighbor, a test on $\mathcal{T}$using $\mathcal{T}$as a training set with k = 1 always gives 100% of training accuracy. It follows, that training accuracy of a plain k-NN with k = 1 will always be greater or equal to the training accuracy of a PML model. It seems that the only way to compare PML and non-PML models is a validation test. This may be hard to be performed on small data.

6.2 Results

7 Conclusions

Unlike the majority of the partial memory systems which belong to the online learning familly, the described in this paper algorithms are batch methods designed to provide an alternative, case-based way of data explanation to the rule discovery analysis. They can also serve as a preprocessing step for large datasets in order to speed up other models. Sufficiently large number of samples should be retained to avoid a 'curse of dimensionality' problem.

Table 1. Results for the 10-fold X 10 CV test on appendicitis data.

System	**Train %**	**Test %**
SBL-PM-B, *k*=1, Euclidean	85.5 $\pm$ 2.4, 3.7 (3.9%)	82.7 $\pm$ 3.0
k-NN, *k*=1, Euclidean	82.6 $\pm$ 2.1, 96 (100%)	82.3 $\pm$ 3.0
SBL-PM-M, *k*=1, Euclidean,	79.1 $\pm$ 2.0, 26.7 (27.8%)	81.4 $\pm$ 3.8

Table 2. Results for the 10-fold X 10 CV test on iris data.

System	**Train %**	**Test %**
SBL-PM-B, *k*=1, Euclidean	96.2 $\pm$ 0.9 4.9 (3.6%)	94.7 $\pm$ 1.7
k-NN, *k*=1, Euclidean	95.4 $\pm$ 0.9 135 (100%)	95.6 $\pm$ 0.6
SBL-PM-M *k*=1, Euclidean	93.4 $\pm$ 1.2, 39.2 (29%)	94.1 $\pm$ 1.4

Table 3. Results for the 10-fold X 10 CV test on promoters data.

System	**Train %**	**Test %**
SBL-PM-B, *k*=1, VDM	90.7 $\pm$ 0.5 22.0 (22.9%)	77.1 $\pm$ 3.3
SBL-PM-M, *k*=1, VDM	90.6 $\pm$ 1.0, 44.8 (47%)	87.4 $\pm$ 1.7
k-NN, *k*=1, VDM	90.2 $\pm$ 0.5 96 (100%)	90.3 $\pm$ 1.6

It is not a surprise that partial memory systems described here usually do not improve and sometimes even degrade the performance of the classifiers employed, as most of the partial memory learning algorithms do. However, after inclusion of them in the similarity based metalearning model [][zacytowac papier ktory ma byc do Machine Learning - w przygotowaniu], they could lead to increase of the prediction ability of the methods based on the SBM framework.

So far only the preliminary numerical experiments had been done with the *k*-nearest neighbors algorithm which had been used as a classification engine. Experiments with other classifiers are in preparation and will be a subject of a separate paper.

What concerns SBL-PM-E, on most datasets there is usually possible to significantly improve over plain *k*-NN method as the results on the test set are excellent. The problem is lack of correlation of classification accuracy computed on training and test sets. This issue will be a subject of further research.

Acknowledgments: Support by Department of Physics, Academy of Bydgoszcz, which made taking part of the author of this paper in this conference possible is gratefully acknowledged.

References

1. W. Duch, *A framework for similarity-based classification methods*, Intelligent Information Systems VII, Malbork, Poland 1998, pp. 288-291