New Generation Computing, 28(2010)21-40
Ohmsha, Ltd. and Springer

An Approach to Data Reduction and Integrated Machine Classification

Ireneusz CZARNOWSKI and Piotr JĘDRZEJOWICZ
Department of Information Systems, Gdynia Maritime University, Morska 83, 81-225 Gdynia, POLAND
irek@am.gdynia.pl, pj@am.gdynia.pl

Received 1 December 2008
Revised manuscript received 28 July 2009

Abstract The goal of the paper is to propose a novel approach to integrated machine classification and to investigate the effect of integration of the data reduction with data mining stage. The integration of both important steps of knowledge discovery in databases is recognized as a vital step towards improving effectiveness of the data mining effort. After having the introduced data reduction and integration schemes a solution to the integrated classification problem is proposed. The proposed algorithm allows for integrating data reduction through simultaneous instance and feature selection, with learning process using population-based and A-Team techniques. To validate the proposed approach and to investigate the effect of data reduction combined with different integration schemes, the computation experiment has been carried out. Experiment based on several benchmark datasets has shown that integrated data reduction and classifier learning outperform traditional approaches.

Keywords: Classification, Machine Learning, Data Mining, Data Reduction, Agent-based Algorithm, Integrated Machine Classification.

§1 Introduction

The amount of data being collected in contemporary databases makes it impossible to reduce and analyze data without the use of automated analysis techniques. Data mining - as a component of the knowledge discovery in databases (KDD), provides techniques for extraction of implicit, unknown, and potentially useful information from data.[12)] Traditionally, within the KDD process, viewed as an iterative sequence of steps, data mining is preceded by data

preprocessing consisting of data collection, data integration, data transformation and cleaning, and, in particular, data reduction.

Typical tasks solved by data mining techniques include the identification of classes (clustering), the prediction of new, unknown objects (classification), the discovery of associations or deviations in spatial databases. Classification is understood as the process of finding a set of models (or functions) that describe and distinguish data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown.[14] The derived model is based on the analysis of a set of training data, which is collection of training examples called instances. For each instance, class label is known. The analysis of the training dataset is usually carried out using machine learning methods. The process of finding a classification model is also referred to as a learning process, supervised learning or learning classifier and the final model is shortly called a classifier. In general, constructing systems that automatically analyze provided examples using machine learning tools is one of the main problems considered in the data mining.

To avoid excessive storage and time complexity, and possibly, to increase learning process quality by avoiding noise and over fitting it is often advisable to reduce original training set by selecting of the relevant data before learning phase or modify the instances using a new representation.[34] The goal of the selection of the relevant data (i.e. data reduction) is to find patterns, also called prototypes or reference vectors, or regularities within certain attributes. Thus, the goal of data reduction approaches is to reduce size of the training dataset without losing extractable information. Such a reduction can result in:

- increasing capabilities and generalization properties of the classification model,
- reducing space complexity of the classification problem,
- decreasing the required computational time,
- diminishing the size of formulas obtained by an induction algorithm on the reduced data sets, and
- speeding up of the knowledge extraction process.[7,21,34]

Data reduction can be achieved by selecting instances and by selecting features/attributes. Instance reduction, often referred to as the selection of reference vectors, becomes especially important in case of large data sets, since overcoming storage and complexity constraints might become computationally very expensive. Although a variety of instance reduction methods has been so far proposed in the literature (see, for example the review,[34] and references [27,32,33]), no single approach can be considered as superior nor guaranteeing satisfactory results in terms of the learning error reduction or increased efficiency of the supervised learning.

Equally important is the reduction of the number of features by removing features that are irrelevant for classification results.[19] In the real-world situations, relevant features are often unknown a priori and, therefore, many features are introduced with a view to better represent the domain.[10] Many of these fea-

tures are not only irrelevant from the point of view of classification results but also can have a negative influence on the accuracy and on the required learning time of the classifier. The presence of redundant features introduces unnecessary noise to the data mining analysis and results in the increased computational complexity of the learning process. As it has been observed in [28], the number of instances needed to assure the required classification accuracy grows exponentially with the number or irrelevant features present. The feature selection problem belongs to the class of NP-hard problems.[22,36]

Another approach to data reduction is based on simultaneously reducing both discussed dimensions of the training dataset. In such case, so called prototypes are found by selecting reference instances and removing irrelevant attributes. The idea of simultaneously reducing both discussed dimensions of the training dataset has been recently investigated in [16,28]. In both papers, the suggested approach is based on using genetic algorithm to train the respective classifiers. In [36], the data reduction problem was combined with the belief revision in the adaptive classification systems. The belief revision is understood as a modification and update of the dataset in two dimensions. Such modification and update of the original dataset is a feature of the adaptive classification systems where the size of the dataset increases in time or where the dataset is huge and a learning is based on a subset of data only.

It is generally accepted that improving quality of machine learning tools can be achieved through integration of both stages of the KDD process i.e. data pre-processing in particular, data reduction with data mining (see for example: [2,5,6,29]). Such an integration may introduce some adaptation mechanisms as exemplified by idea of learning classifier systems.[6,17] Integration of the preprocessing and classifier construction stages leads however to a considerable extension of the decision space at the learning process. Both - feature selection and instance selection, are computationally difficult (see [22,28]) and of course the resulting problem of constructing the integrated classifier is no less computationally difficult.

The goal of the paper is to propose a new approach to data reduction. The idea is to design and implement an agent-based population learning algorithm for data reduction. The motivation is the authors' belief that there is still a room for improving data reduction techniques. The goal of the paper is to show that the agent-based population learning algorithm allows for integration of the data reduction process with learning process resulting in the increased performance of machine learning tools and quality of the data mining process.

The proposed agent-based approach to solving the problem of deriving the classification model by integrating data reduction and data mining stages uses advantages of the A-team approach [31] with inherent adaptation mechanism. The idea of the agent-based population learning algorithm has been presented in [4], where it has been demonstrated that the agent-based population algorithm can be successfully used to obtain good quality solutions of NP-hard problems.

The paper is organized as follows. Section 2 reviews briefly data reduction algorithms. The integrated and adaptive classification is discussed in Section 3.

Section 4 provides details of the proposed approach. To validate it the computational experiment has been planned and carried out. Its results are presented and discussed in Section 5. Finally, in the last section, some conclusions are drawn and directions of future research are suggested.

§2 Data Reduction Algorithms

2.1 Selection of Reference Instances

In general, instance reduction problem concerns removing a number of instances from the original training set T and thus producing the reduced training set S. Usually, instance reduction algorithms are based on a distance calculation between instances in the training set. In such case, selected instances, which are situated close to the center of clusters of similar instances, serve as the reference instances.[8)] The approach requires using some clustering algorithms. Other approaches, known as instance-based methods, remove an instance if it has the same output class as its k nearest neighbors assuming that all neighborhood instances will be, after all, correctly classified. Instance-based methods must, in general, decide what examples to store for use in the learning process. However, these methods have several weaknesses. They very often use a single distance function that can be inappropriate for different types of attributes (for example, linear and nominal ones). Besides, there is a needed to store all of the available training examples in the model. Furthermore, the discussed methods are, in general, very slow during the execution, because all of the training instances must be reviewed in order to classify each new instance. These and others weaknesses of the nearest neighbor algorithm and others based on a distance classifications are widely discussed in [33)]. To eliminate the above weaknesses of the nearest neighbor algorithm many approaches has been proposed, including CNN,[15)] IB2 and IB3,[1)] SNN,[27)] ENN [34)] and DROP1-DROP5.[34)] The other group of methods (i.e. for example IRA1-IRA2, All k-NN) try to eliminate unwanted training examples basing on some removal criteria that need to be fulfilled.[9,32)] Instance reduction problem, often referred to as the prototype selection problem, can be also solved by heuristics, random or evolutionary search.[7,28,30)] The Wilson and Martinez in [34)] suggested that the search for the reduced training set S can be carried out in one of the three modes, incremental, decremental and batch.

In the literature, one can also find others approaches to instance selection. For example, in [20)] and [24)], the technique for data reduction based on the idea of sampling has been proposed. Sampling methods start with small number of instances in the initial iteration and than boost the training dataset until model accuracy no longer improves. Another approach, called stratification strategy, derives, randomly, the initial dataset into a set of strata. In next step prototypes from each strata are selected. In this case, the number of strata is a parameter of the strategy.[7)]

In [9)], the instance reduction is achieved through calculating, for each instance from the original set, the value of its similarity coefficient, and then grouping instances into clusters consisting of instances with identical values of

this coefficient, selecting the representation of instances for each cluster and removing the remaining instances. Finally, the selected reference vectors produce the reduced training set. In this case, clusters, corresponding to strata, are created based on similarity between instances. In [9], it was shown that such an approach can result in reducing the number of instances and still preserving a quality of the data mining results. It was also demonstrated that, in some cases, reducing the training set size can increase efficiency of the supervised learning.

The data reduction approach, proposed in this paper, uses to identify clusters of instances values of the similarity coefficient calculated as in [9]. After grouping instances into clusters with identical values of the similarity coefficients, prototypes are selected by the agent-based population learning algorithm.

To present the proposed approach in a formal manner the following notation needs to be introduced. Let x denote a training example, N denote the number of instances in the original training set T and n - the number of attributes. Total length of each instance (i.e. training example) is equal to $n+1$, where element numbered $n+1$ contains the class label. The class label of each example can take any value from a finite set of decision classes $C = \{c_l : l = 1, \ldots, k\}$, which has cardinality k. Also, let $X = \{x_{ij}\}$ $(i = 1, \ldots, N; j = 1, \ldots, n+1)$ denote the matrix of $n+1$ columns and N rows containing values of all instances from T. The detailed pseudo-code of the procedure producing clusters of instances is shown below as Algorithm 1.

***Algorithm 1**: The instance grouping procedure*

Input: X - the matrix containing values of all instances from T.
Output: $Y_1, \ldots, Y_t$, where t is the number of clusters consisting of instances with identical value of the similarity coefficient.
1. Transform data instances: each $\{x_{ij}\}$ for $i = 1, \ldots, N$ and $j = 1, \ldots, n$ is normalized into interval $[0, 1]$ and then rounded to the nearest integer,
that is 0 or 1.
2. Calculate values:

$$s_j = \sum_{i=1}^{N} x_{ij}, \quad \text{where} \quad j = 1, \ldots, n. \tag{1}$$

3. For instances from X, belonging to the class c_l (where $l = 1, \ldots, k$), calculate the value of its similarity coefficient I_i:

$$\forall_{x: x_{i,n+1} = c_l} \; I_i = \sum_{j=1}^{n} x_{ij} s_j, \quad \text{where} \quad i \in \{1, 2, \ldots, N\}. \tag{2}$$

4. Map input vectors from X with the same value of similarity coefficient I_i into clusters. Let $Y_1, \ldots, Y_t$ denote the obtained clusters.

Thus, the above procedure groups instances from T, based on its similarity, into t disjoint subsets $Y_1, \ldots, Y_t$ such that $T = \bigcup_{i=1}^{t} Y_i$ and $Y_i \cap Y_j = \emptyset$ $\forall i \neq j$. Next from these subsets the reference instances could be selected and the

reduced training set S could be produced, where initially $S = \emptyset$. The selection is based on the following rules:

- If $|Y_i| = 1$ then $S := S \cup Y_i$, where $i = 1, \ldots, t$.
- If $|Y_i| > 1$ then $S := S \cup \{x^i\}$, where x^i is a reference instance selected from the cluster Y_i. In this paper, this reference vector is selected by the agent-based population learning algorithm described in Section 4.

Applying the above algorithm can result is substantial reduction of training set size as compared with the original data set. We claim that reducing training set can still preserve features of the analysed data. Intuitively, this can be observed in case of the Example 1.

Example 1: In Fig. 1, the distribution of values of the sepal length and width from the Iris problem from the UCI Machine Learning Repository [3)] consisting originally of 150 instances is shown and compared with their distribution obtained after applying the Algorithm 1 and reducing the number of instances from 150 to 11.

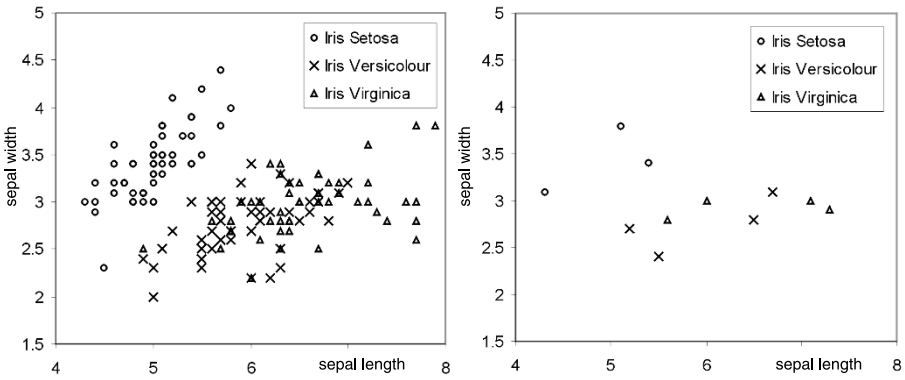

Fig. 1 Initial distribution of values of attributes "sepal length" and "sepal width" from the Iris problem (on the left) and their final distribution after reduction of the number of instances (on the right).

2.2 Feature Selection

Equally important, as the selection of reference instances, is the reduction of the number of features, also known as feature selection, variable selection, feature reduction, attribute selection or variable subset selection. Feature selection is the technique, commonly used in machine learning, of selecting a subset of relevant features for building robust learning models. The goal of the feature selection is a reduction of the search space, by selection of relevant features and removal of the remaining irrelevant features. That is, feature selection selects m features from the entire set of n features such that $m < n$. Ideally $m <<< n$.

The problem of finding the optimal subset of features is NP-hard.[36] Feature selection method based on enumeration and review of all subset of features leads directly to finding the best one. However, this procedure is very time consuming and in majority of cases not feasible. The statistical methods evaluate independently each attribute. Other approaches attempt to evaluate and rank some subsets of features that were generated by heuristics, random or genetic search procedures. All feature selection methods that have been so far proposed, try to find the best subset of features satisfying certain criteria. Among them are: the classifier error rate, information entropy, dependence measure, consistency measure or distance measure. The discussion of the feature selection approaches can be found in [10, 19, 26, 38].

§3 Integrated and Adaptive Classification

As it was pointed in Section 1, integration of the data reduction with the learning classifier has been recognized as a promising step towards increasing effectiveness of the KDD process. Such an integration allows also to introduce same adaptation mechanisms into the KDD process. An integrated adaptive classifiers should possess two important features:

- Integration, at least partial, of data reduction and data mining stages.
- The existence of a positive feedback whereby more effective data reduction leads to a higher classification accuracy, and in return, higher classification accuracy results in even more effective data reduction.

Within the traditional model of the KDD, feature and instance selection processes are not integrated with the learning classifier. Training set thus produced forms approximation space, which is not adaptable. In another words, approximation space used for the purpose of the classifier construction remains constant. In such case, the machine classification problem can be defined as follows: given the approximation space in the form of the training set produced at the pre-processing stage, construct classifier which performs best from the point of view of the performance criterion used.

Situation changes with the introduction of the integrated and adaptive classifier. In such case, some processes within the pre-processing stage, like instance selection and/or feature selection can be integrated with the construction of the classifier which allows for introducing adaptability of the approximation space. Thus the machine classification problem can be formulated as follows: construct classifier which performs best from the point of view of the performance criterion by finding the representation of the approximation space and, at the same time, deciding on the classifier features. Unfortunately, integration of the pre-processing and classifier construction stages leads to a considerable extension of the decision space at the classifier construction phase.

Based on the above review of the instance and feature selection techniques, the following integration schemes can be considered:

- Integration of the instance selection with the learning classifier stage - the

approach allows to construct a classifier and, at the same time, to modify the approximation space obtained from the training set by changing the way instances are represented or simply by selecting instances. Modified approximation space allows to improve the classifier performance. Additionally, such schema allows others data improving mechanisms, like feature selection, at the pre-processing stage.
- Integration of feature selection with the learning classifier stage - this allows to construct the classifier and, at same time, to modify the approximation space obtained from the training set by selecting a subset of features. Modified approximation space allows to improve the classifier performance. Additionally, this schema allows others data improving mechanisms, like instance selection, at the pre-processing stage.
- Integration of the both - instance and feature selection, with the learning classifier stage - instances and features are selected iteratively while constructing the classifier.

§4 A Solution of the Integrated and Adaptive Classification Problem

4.1 Main Features of the Proposed Approach

Since 1976 when Holland proposed Learning Classifier Systems,[17] the idea of applying machine learning techniques which combine evolutionary computing, reinforcement learning, supervised learning and heuristics to produce adaptive systems have been gaining a growing interest from the machine intelligence research community. The approach proposed in this paper integrates features of the evolutionary computation,[23] local search algorithms,[13] population learning algorithm,[18] multiple-agent systems and the A-Team concept.[31]

The proposed solution provides instance and feature reduction capabilities integrated with the learning classifier process. The idea is to build and use an A-Team structure where multiple agents search for the best combination of features and instances using local search heuristics and population based methods. The best combination of features the A-Team is searching for, is selected from the population of potential solutions which are kept in the common memory. Specialized agents try to improve solutions from the common memory by changing values of the decision variables (either selected instances or features). All agents can work asynchronously and in parallel. Agents cooperate to construct or find a solution by selecting and modifying solutions, which are drawn from the shared common memory and store them back after attempted improvement. Their interactions provide for the required adaptation capabilities and for the evolution of the population of potential solutions.

In our case, the shared memory is used to store a population of solutions. Each solution is represented by the set of prototypes i.e. by the compact representation of the data set. The team of agents is used to find the best solution.

4.2 Solution Representation

Population of solutions to the considered problem consists of feasible solutions. A feasible solution s, which is a set of selected data, is represented by a string consisting of numbers of selected reference instances and numbers of selected features. The first t numbers represent instance numbers from the reduced data set T, where t is a number of clusters of potential reference instances. The value of t is calculated at the initial population generation phase, where at first, the clusters of instances are identified through applying the Algorithm 1. Thus, the first t numbers refer to the respective reference instances from each cluster. In such a representation, each cluster is represented by a single instance. The second part of the solution is a string containing numbers of the selected features. It has a length of minimum 1 and maximum n. The example of the feasible solution is shown below.

Example 2: The number of instances in the original data set is 15 numbered from 1 to 15, and there are 6 attributes numbered from 1 to 6. Let the considered string be: $s = [4, 7, 12, 1, 2, 4, 5, 6]$ with $t = 3$, where t determines the number of clusters in s, thus:

- the first 3 position include numbers of the reference instances that represent three clusters,
- $s_1 = 4$ is the number of vector selected to represent the first cluster,
- $s_2 = 7$ is the number of vector selected to represent the second cluster,
- $s_3 = 12$ is the number of vector selected to represent the third cluster and
- s_i, for $i \in \{4, 5, 6, 7, 8\}$, represents number of the selected feature. In the example, features numbered 1, 2, 4, 5 and 6 are selected.

4.3 Agent-based Search

The proposed solution, based on a dedicated team of agents, has been denoted as *IAC* (*Integrated and Adaptive Classification*). The *IAC* functionality is realized by two main types of agents. The first one - *optimizing agents*, are implementation of the improvement algorithms, which aim at improving the current solution. Each *optimizing agent* represents a single improvement algorithm. The second one, called as *solution manager*, is responsible for management of the population of solution and updating individuals in the population. The *optimizing agents* cooperate to find a better solution with the *solution manager* and work in parallel.

The *solution manager* role is to manage the population of solutions, which, at the initial phase, is generated randomly and stored in the shared memory. It means, that at the initial phase potential solutions are generated by random selection of a single instance from each cluster and by random selection of feature numbers. It also means, that before the initial phase the *solution manager* runs Algorithm 1 to obtain clusters with instances having identical value of the similarity coefficient. Such a procedure is expected to assure the required diversity within the initial population of solutions.

After the initialization phase the *solution manager* continues reading in-

dividuals from the common memory and storing them back after attempted improvement until a stopping criterion is met. During this process, the *solution manager* keeps sending single individuals (solutions) from the common memory to *optimizing agents*. Solutions forwarded to *optimizing agents* are drawn randomly. Each *optimizing agent* tries to improve quality of the received solutions and afterwards sends them back to the *solution manager*, which, in turn, updates common memory by replacing a randomly selected or worst individual (depending on the preferred evolution strategy) with the improved one. The whole process continues until some stoping criteria is met.

To solve the data reduction problem four types of optimizing agents carrying out different improvement procedures have been implemented. The procedures used by respective agents are shown as algorithms 2, 3, 4 and 5.

***Algorithm 2**: Local search with tabu list for instance selection*

Input: s - individual representing a solution encoded as a string defined in Section 4.2; L - list of the problem instance numbers not in s; t - number of clusters in s; $T = \emptyset$ - tabu list; x - number of iterations an instance number stays on the tabu list.
Output: *solution* - the improved individual encoded as a string defined in Section 4.2.
1. Set k by drawing it at random from $\{1, 2, .., t\}$
2. Identify r which is an instance number representing the k^{th} cluster
3. If ($r \in T$) then goto 9.
4. Set r' by drawing it at random from L
5. Replace an instance numbered r by an instance numbered r' within the k^{th} cluster of s thus producing individual s'
6. Calculate fitness of s'
7. If (s' is better then s) then ($s := s'$ AND r replaces r' in L AND r is added to T)
8. Remove from T instances staying there for x iterations
9. If (!terminating condition) then goto 1.
10. *solution* $:= s$

***Algorithm 3**: Local search for instance selection*
Input: s - individual representing a solution encoded as a string defined in Section 4.2; L - list of the problem instance numbers not in s; t - number of clusters in s.
Output: *solution* - the improved individual encoded as a string defined in Section 4.2.
1. Set k by drawing it at random from $\{1, 2, .., t\}$
2. Identify r which is an instance number representing the k^{th} cluster
3. Set r' by drawing it at random from L
4. Replace an instance numbered r by an instance numbered r' within the k^{th} cluster of s thus producing individual s'
5. Calculate fitness of s'
6. If (s' is better then s) then ($s := s'$ AND r replaces r' in L)
7. If (!terminating condition) then goto 1.
8. *solution* $:= s$

Algorithm 4*: Local search with tabu list for feature selection*
Input: s - individual representing a solution encoded as a string defined in Section 4.2; M - list of the feature numbers in s; M' - list of the feature numbers not in s; $T = \emptyset$ - the tabu list; x - number of iterations a feature number stays on the tabu list.
Output: *solution* - the improved individual encoded as a string defined in Section 4.2.
1. Set f by drawing it at random from M
2. If ($f \in T$) then goto 8.
3. Set f' by drawing it at random from M'
4. Replace f by f' in s thus producing s'
5. Calculate fitness of s'
6. If (s' is better then s) then ($s := s'$ AND update M and M' AND add f to T)
7. Remove from T features staying there for x iterations
8. If (!terminating condition) then goto 1.
9. *solution* := s

Algorithm 5*: Local search for instance and feature selection*
Input: s - individual representing a solution encoded as a string defined in Section 4.2; L - list of the problem instance numbers not in s; M - list of the feature numbers in s; M' - list of the feature numbers not in s.
Output: *solution* - the improved individual encoded as a string defined in Section 4.2.
1. Set i:=0
2. Set h by drawing it at random from $\{0, 1\}$
3. If ((i *mod* h) is not 0) then goto 14.
4. Set d by drawing it at random from $\{0, 1\}$
5. If (d is not 0) then goto 10.
6. Set f' by drawing it at random from M'
7. Add f' to s producing s'
8. Calculate fitness of s'
9. If (s' is better then s) then ($s := s'$ AND update M and M' AND goto 20.)
10. Set f by drawing it at random from M
11. Remove f from s producing s'
12. Calculate fitness of s'
13. If (s' is better then s) then ($s := s'$ AND update M and M' AND goto 20.)
14. Set k by drawing it at random from $\{1, 2, .., t\}$
15. Identify r which is an instance number representing the k^{th} cluster
16. Set r' by drawing it at random from L
17. Replace an instance numbered r by an instance numbered r' within the k^{th} cluster of s thus producing individual s'
18. Calculate fitness of s'
19. If (s' is better then s) then ($s := s'$ AND r replaces r' in L)
20. If (!terminating condition) then ($i := i + 1$ AND goto 3.)
21. *solution* := s

In each of the above cases, the modified solution replaces the current one if it is evaluated as a better one. Evaluation of the solution is carried out by

estimating classification accuracy of the classifier, which is constructed taking into account the instances and features as indicated by the solution.

If, during its search, an agent successfully has improved the received solution, then it stops and the improved solution is transmitted to the *solution manager*. Otherwise, agent stops searching for an improvement after having completed the prescribed number of iterations.

From the technical point of view, the *IAC* was implemented using JADE-Based A-Team (JABAT) environment. JABAT, based on JAVA code, is a middleware supporting the construction of the dedicated A-Team architecture that can be used for solving a variety of computationally hard optimization problems. A detailed description of JABAT can be found in [4].

§5 Computational Experiment Results

To evaluate the performance of the proposed algorithm, it has been decided to carry out the computational experiment. The aim of the experiment was to evaluate to what extend the proposed approach could contribute towards increasing classification accuracy of classifier induced from the set of prototypes selected by applying an agent-based population learning algorithm with the inherent adaptation mechanism. The aim of the experiment was also to compare the performance of different integration schemes. Classification accuracy of the classifier obtained using the proposed approach (i.e. using the set of prototypes, found by simultaneously selecting reference instances and removing irrelevant attributes, and by integration of data reduction with learning model) has been compared with:

- results obtained by machine classification without the integrated data reduction, that is using full dataset,
- results obtained by machine classification with the different integration and reduction schemes.

To validate the proposed approach several benchmark classification problems have been solved. Datasets for each problem have been obtained from the UCI Machine Learning Repository.[3] They include: Cleveland heart disease (303 instances, 13 attributes, 2 classes), credit approval (690, 15, 2), Wisconsin breast cancer (699, 9, 2) and sonar problem (208, 60, 2).

Each benchmarking problem has been solved 30 times and the reported values of the quality measures have been averaged over all runs. The quality measure in all cases was the correct classification ratio calculated using the 10-cross-validation approach. All optimization agents have been allowed to continue iterating until 100 iterations have been performed. The common memory size was set to 100 individuals. The number of iterations and the size of common memory have been set out experimentally at the fine-tuning phase. The whole process of searching for the best solution stops when there are no improvements of the best solution during the last 3 minutes of computation.

Experiment results are shown in Table 1. These results have been obtained by using the: C 4.5 classifier without pruned leaves and with pruned

Table 1 Accuracy (%) of Classification Results Obtained for Different Selected Learning Schemes

Classifier / Problem	1NN	10NN	Bayes Network	WLSVM	C 4.5 (pruned)	C 4.5 (unpruned)
	Case A: no data reduction applied					
cancer	95.71	96.71	96.00	**95.57**	94.57	95.00
credit	82.46	86.38	75.36	85.22	84.93	83.19
heart	77.23	80.86	83.50	80.53	77.89	76.90
sonar	94.23	75.00	73.08	**72.12**	74.04	74.04
average	87.41	84.74	81.98	83.36	82.86	82.28
	Case B: feature selection at the pre-processing stage					
cancer	94.57	95.00	96.00	95.14	94.43	94.43
credit	79.57	84.64	84.93	85.51	77.25	74.78
heart	71.95	80.53	80.53	80.86	79.87	79.54
sonar	79.33	81.25	61.54	73.08	72.10	71.23
average	81.35	85.35	80.75	83.65	80.91	79.99
	Case C: integrated instance selection, no feature selection					
cancer	96.86	97.43	96.87	90.59	97.44	**98.43**
credit	83.33	88.70	75.22	**85.94**	90.72	90.43
heart	84.00	85.67	87.33	87.00	91.21	**92.42**
sonar	94.23	75.00	75.00	40.38	83.65	83.65
average	89.61	86.70	83.61	75.98	90.76	91.24
	Case D: integrated instance selection, feature selection carried at the pre-processing stage					
cancer	94.15	96.15	96.72	74.49	95.15	95.44
credit	75.22	72.61	77.54	64.49	81.88	81.16
heart	83.33	87.00	86.00	87.00	85.00	85.67
sonar	76.32	72.31	63.56	71.23	82.02	81.72
average	82.26	82.02	80.95	74.30	86.01	86.00
	Case E: integrated feature selection, instance selection carried at the pre-processing stage					
cancer	89.31	94.73	75.16	73.64	95.01	86.45
credit	69.28	61.30	69.71	59.57	81.32	76.71
heart	75.04	81.33	76.00	83.00	81.67	81.33
sonar	74.02	79.32	71.65	65.86	78.43	76.43
average	76.91	79.17	73.13	70.52	84.11	80.23
	Case F: integrated instance and feature selection					
cancer	**98.15**	**98.29**	**98.15**	88.01	**98.15**	97.94
credit	**87.68**	**89.57**	**85.22**	85.51	**92.61**	**90.87**
heart	**88.67**	**89.33**	**89.33**	**90.00**	**93.00**	92.17
sonar	**95.19**	**82.69**	**80.77**	72.08	**87.50**	**88.85**
average	92.42	89.97	88.37	83.90	92.81	92.45

leaves,[25] support vector machine (WLSVM) [37] and 1NN, kNN, Bayes Network implemented in WEKA library.[35] Further, the results in Table 1 are presented for the following cases:

- Case A: results obtained by machine classification using the full dataset,
- Case B: results obtained by machine classification with feature selection at the pre-processing stage,
- Case C: results obtained by machine classification with the integrated instance selection and without feature selection at the pre-processing stage,
- Case D: results obtained by machine classification with the integrated instance selection and with feature selection carried out at the pre-processing stage,
- Case E: results obtained by machine classification with the integrated feature selection and with instance selection carried out at the pre-processing stage,
- Case F: results obtained by machine classification with the integrated instance and feature selection.

In all cases shown in Table 1, the non-integrated feature selection was carried out at the pre-processing stage using the *wrapper* technique.[10)] The non-integrated instance selection was based on selection of reference vectors as proposed in [9)]. In all the remaining cases, the results shown in Table 1 were obtained applying the proposed approach described in Section 4.

From Table 1, it is clear that the proposed agent-based integrated data reduction technique is competitive with respect to classification accuracy in comparison with the traditional approach. Moreover, it is also competitive if compared with other classifiers and other approaches to data reduction. This can be concluded from data shown in Table 2 where the results obtained by using the fully integrated approach (case F in Table 1) are compared with the best classification results reported in the literature. The column $|S|/|T|$ in Table 2 shows what percentage of instances from the original training set has been retained by the respective reduction algorithm. The column *type* in Table 2 indicates a type of data reduction applied. In particular, this column contains the following notation:

"I" - the data reduction has been carried out applying only instance selection,
"F" - the data reduction has been carried out applying only feature selection,
"IF" - the data reduction has been carried out applying simultaneous instance and feature selection,
"N" - no data reduction applied.

Results obtained during the experiment and presented in Tables 1 & 2 show that the integration of both that is instance and feature reduction, with learning classifier assures a very good quality of solutions in case of the analyzed benchmark classification problems.

It can be observed that the proposed approach guarantees also quite a satisfactory reduction of the original dataset. In case of all of the considered problems, integrating instance and feature selection with the learning process have increased the classification accuracy as compared with accuracy obtained by training the classifier using the original full dataset. This observation holds true

Table 2 Performance Comparison of Different Classifiers and Instance Reduction Algorithms

Approach		cancer		heart		credit		sonar	
	type	Accur.	$\frac{\lvert S\rvert}{\lvert T\rvert}$	Accur.	$\frac{\lvert S\rvert}{\lvert T\rvert}$	Accur.	$\frac{\lvert S\rvert}{\lvert T\rvert}$	Accur.	$\frac{\lvert S\rvert}{\lvert T\rvert}$
IAC+C4.5	IF	98.15%	20%	93.0%	60%	92.61%	30%	8758%	90%
IAC+10NN	IF	98.29%	20%	89.33%	60%	89.57%	30%	82.69%	90%
IAC+1NN	IF	98.15%	20%	88.67%	60%	87.68%	30%	95.19%	90%
IAC+WLSVM	IF	95.57%	20%	80.53%	60%	85.22%	30%	72.12%	90%
IAC+BayesNet.	IF	96.0%	20%	83.5%	60%	75.36%	30%	73.08%	90%
k-NN [34]	N	96.28%	100%	81.19%	100%	84.78%	100%	58.8%	100%
CNN [34]	I	95.71%	7.09%	73.95%	30.84%	77.68%	24.22%	74.12%	32.85%
SNN [34]	I	93.85%	8.35%	76.25%	33.88%	81.31%	28.38%	79.81%	28.26%
IB2 [34]	I	95.71%	7.09%	73.96%	30.29%	78.26%	24.15%	80.88%	33.87%
IB3 [34]	I	96.57%	3.47%	81.16%	11.11%	85.22%	4.78%	69.38%	12.02%
DROP3 [34]	I	96.14%	3.58%	80.84%	12.76%	83.91%	5.96%	78%	26.87%
RMHC [30]	IF	70.9%	7%	82.3%	3%	-	-	-	-
GA-KJ [28]	IF	95.5%	33.4%	74.7%	33.1%	-	-	55.3%	52.6%
1NN+RELIEF [26]	F	72.12%	100%	77.85%	100%	79.57%	100%	-	-
IB3+RELIEF [26]	F	73.25%	100%	79.94%	100%	71.75%	100%	-	-
ID3+FSS [19]	F	94.53%	100%	-	-	-	-	-	-
ID3 [11]	N	94.3%	100%	-	-	-	-	-	-
C 4.5+BFS [11]	F	95.28%	100%	-	-	-	-	-	-
C 4.5 [16]	N	94.7%	100%	77.8%	100%	85.5%	100%	76.9%	100%
BayesNet. [11]	N	96.7%	100%	83.6%	100%	82.9%	100%	90.4%	100%
SVM [11]	N	96.9%	100%	81.5%	100%	-	-	76.9%	100%
MLP+BP [11]	N	96.7%	100%	81.3%	100%	84.6%	100%	90.4%	100%

independently from the machine learning tool. Gains in classification accuracy seem to be quite substantial.

For example, classifiers based on the original training sets produce for the set of investigated benchmark problems average classification accuracy of 82.86% and 82.28%, using C 4.5 with pruned and unpruned leaves, respectively, while the proposed approach to integration instance and feature reduction with learning classifier assures average accuracy of 92.81% and 92.45%, respectively. In case of the *sonar* problem, considered to be a difficult one, the improvement is even more spectacular (from the average accuracy of 74.04% to the accuracy of 88.85%, for C 4.5, when integration instance and feature selection with classifier construction is applied). The experiment results also show that the proposed approach to feature reduction results in better classification accuracy as compared to accuracy obtained through applying the wrapper technique.

To reinforce our conclusions, the experiment results, shown in Table 1, have been used to perform the two-way analysis of variance. The following null hypothesis were formulated:

I. Choice of the integration scheme does not influence the classifier performance.
II. Choice of the classifier type does not influence the classification accuracy.
III. There are no interactions between both factors (i.e. choice of the integration scheme and choice of the classifier type).

Table 3 The ANOVA F Test Values Summary Table

Problem	*df*	credit	cancer	heart	sonar	F_{crit}
Main effect A	5	5.93683	6.86430	5.16468	7.08378	2.22267
Main effect B	5	1.62890	1.93260	2.03260	1.95232	2.22267
Interaction effect	20	0.46085	0.70185	0.42165	0.23422	1.51666

Table 4 Average Number of Rules and Average Size of the Decision Tree

Algorithm	C 4.5 - pruned leaves					C 4.5 - unpruned leaves				
Case	A	B	C	D	E	A	B	C	D	E
Problem	Average number of rules									
credit	12.0	36.0	16.4	15.5	10.5	54.0	75.0	24.5	15.4	13.9
cancer	15.0	8.0	8.2	2.3	6.5	20.0	19.0	12.8	2.7	7.7
heart	17.0	11.0	17.6	8.7	11.5	44.0	26.0	23.8	8.4	14.3
sonar	8.0	10.0	9.0	16.0	11.0	8.0	12.0	10.0	14.0	11.4
	Average size of the tree									
credit	23.0	71.0	31.8	30.0	20.0	107.0	149.0	48.0	29.8	26.8
cancer	29.0	15.0	15.4	3.6	12.0	39.0	37.0	24.6	4.4	14.3
heart	33.0	21.0	34.3	16.4	22.0	87.0	35.0	46.6	15.8	27.6
sonar	15.0	20.0	17.0	19.0	21.0	15.0	25.0	19.0	17.0	21.8

It was established that with the degree of confidence set at 95% hypothesis II. and III. hold true. However, hypothesis I. should be rejected. Table 3 includes, as the analyzis of variance summary, the ANOVA F test values. In addition, Tukey test confirmed that in case of the integrated approach, there are no statistically significant differences between mean performances obtained using different classifiers.

On the other hand, the experiment results show that the data reduction can result in a decreased complexity and size of the decision tree as compared with the decision tree constructed using a full, non-reduced dataset. This can be concluded from results presented in Table 4, where average number of rules and average size of the decision tree are shown. It is clear that the proposed approach results in a decreasing complexity of knowledge representation as well as in a reduction of computation time required. This remains true not only for decision trees but also for other machine learning techniques.

§6 Conclusions

Main contribution of the paper is proposing and validating an approach to the integrated machine classification where data reduction in both - feature and instance dimension is carried out iteratively and in parallel with constructing a classifier. The proposed solution has been implemented using the multi-agent platform. Main principles behind such an approach include population-based computations, A-team based search for optimal solutions and the proposed similarity factor as a tool for selecting reference instances. Combining the above into an adaptive and distributed classification system has resulted in constructing an effective and dependable classification tool. Validating experiment results enable to draw the following further conclusions:

- reducing training set size still preserves knowledge content of the analyzed

data,
- integration of the preprocessing stage with learning classifier assures better results then a non-integrated solution,
- choice of the data integration scheme may significantly influence classifier performance,
- choice of the classifier is not a decisive factor from the point of view of performance of the integrated classifier.

The proposed technique extends the range of available approaches to classifier construction. Moreover, it is shown that the proposed algorithm can be, for some problems, competitive in comparison with other existing techniques. Properties of the proposed algorithm should be further studied. Future research will also aim at refining the available data reduction techniques and extending the approach to the distributed datasets.

Acknowledgements

This research has been supported by the Polish Ministry of Science and Higher Education with grant for years 2008-2010.

References

1) Aha, D.W., Kibler, D., Albert, M.K., "Instance-based learning algorithms," *Machine Learning, 6*, pp. 37-66, 1991.
2) Aksela, M., *Adaptive Combinations of Classifiers with Application to On-line Handwritten Character Recognition*, Ph.D. Thesis, Department of Computer Science and Engineering, Helsinki University of Technology, Helsinki, 2007.
3) Asuncion, A., Newman, D.J., "UCI Machine Learning Repository," Irvine, CA: University of California, School of Information and Computer Science, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 2007.
4) Barbucha, D., Czarnowski, I., Jędrzejowicz, P., Ratajczak-Ropel, E. and Wierzbowska, I., "JADE-Based A-Team as a Tool for Implementing Population-Based Algorithms," in *Proc. of the Sixth International Conference on Intelligent Systems Design and Applications (ISDA'06)* (Yuehui Chen, Ajith Abraham eds.) IEEE Computer Society, *3*, pp. 144-149, 2006.
5) Bhanu, B., Peng, J., "Adaptive Integration Image Segmentation and Object Recognition," *IEEE Trans. on Systems, Man and Cybernetics, 30, 4*, pp. 427-441, 2000.
6) Bull, L., "Learning Classifier Systems: A Brief Introduction," in*Applications of Learning Classifier Systems, Studies in Fuzziness and Soft Computing* (Larry Bull ed.), Springer, 2004.
7) Cano, J. R., Herrera, F., Lozano, M., "On the Combination of Evolutionary Algorithms and Stratified Strategies for Training Set Selection in Data Mining," in *Pattern Recognition Letters*, Elsevier, 2004.
8) Chang, C.-L., "Finding Prototypes for Nearest Neighbor Classifier," *IEEE Transactions on Computers, 23, 11*, pp. 1179-1184, 1974.

9) Czarnowski, I., Jędrzejowicz, P., "An Approach to Instance Reduction in Supervised Learning," in *Research and Development in Intelligent Systems XX* (Coenen F., Preece A. and Macintosh A. eds.), Springer, London, pp. 267-282, 2004.

10) Dash, M., Liu, H., "Feature Selection for Classification," *Intelligence Data Analysis, 1, 3*, pp. 131-156, 1997.

11) Duch, W., "Results - Comparison of Classification," Nicolaus Copernicus University, http://www.is.umk.pl/projects/datasets.html, 2002.

12) Frawley, W. J., Piatetsky-Shapiro, G. and Matheus, C., "Knowledge Discovery in Databases - An Overview," in *Knowledge Discovery in Databases* (Piatetsky-Shapiro G., Matheus C. eds.) AAAI/MIT Press, 1991.

13) Glover, F., "Tabu search. Part I and II," *ORSA Journal of Computing, 1, 3*, Summer 1990 and *2*, *1*, Winter 1990.

14) Han, J., Kamber, M., *Data Mining. Concepts and Techniques*, Academic Press, San Diego, 2001.

15) Hart, P. E., "The Condensed Nearest Neighbor Rule," *IEEE Transactions on Information Theory, 14*, pp. 515-516, 1968.

16) Ishibuchi, H., Nakashima, T. and Nii, M., "Learning of Neural Networks with GA-based Instance Selection," in *Proc. of the IFSA World Congress and 20th NAFIPS International Conference, 4*, pp. 2102-2107, 2001.

17) Holland, J. H., "Adaptation," in *Progress in Theoretical Biology* (Rosen & Snell eds.), *4*, Plenum, 1976.

18) Jędrzejowicz, P., "Social Learning Algorithm as a Tool for Solving Some Difficult Scheduling Problems," *Foundation of Computing and Decision Sciences, 24*, pp. 51-66, 1999.

19) Kohavi, R., John, G. H., "Wrappers for Feature Subset Selection," *Artificial Intelligence , 97, 1-2*, pp. 273-324, 1997.

20) Lazarevic, A., Obradovic, Z., "Data Reduction Using Multiple Models Integration," in *Proc. of the 5th European Conference Principles and Practice of Knowledge Discovery and Databases, LNCS 2168*, Springer-Verlag London, pp. 301-313, 2001.

21) Liu, H., Lu, H., Yao, J., "Identifying relevant databases for multidatabase mining," in *Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 210-221, 1998.

22) Meiri, R., Zahavi, J., "Using Simulated Annealing to Optimize the Feature Selection Problem in Marketing Applications," *European Journal of Operational Research, 17, 3*, pp. 842-858, 2006.

23) Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin, 1996.

24) Morgan, J., Daugherty, R., Hilchie, A., Carey B., "Sample size and modeling accuracy of decision tree based data mining tools," *Academy of Information and Management Science Journal, 6, 2*, pp. 71-99, 2003.

25) Quinlan, J. R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, 1993.

26) Raman, B., Ioerger, T. R., "Enhancing learning using feature and example selection," in *Journal of Machine Learning Research*, 2003.

27) Ritter, G. L., Woodruff, H. B., Lowry, S. R., Isenhour, T. L., "An Algorithm for a Selective Nearest Decision Rule," *IEEE Trans. on Information Theory, 21*, pp. 665-669, 1975.

28) Rozsypal, A., Kubat, M., "Selecting Representative Examples and Attributes by a Genetic Algorithm," *Intelligent Data Analysis, 7, 4*, pp. 291-304, 2003.

29) Sahel, Z., Bouchachia, A., Gabrys, B., Rogers P., "Adaptive Mechanisms for Classification Problems with Drifting Data." in *KES 2007* (B. Apolloni et al. eds.), *LNAI 4693*, Springer-Verlag, pp. 419-426, 2007.

30) Skalak, D. B., "Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithm," in *Proc. of the International Conference on Machine Learning*, pp. 293-301, 1994.

31) Talukdar, S., Baerentzen, L., Gove, A. and De Souza, P., "Asynchronous Teams: Cooperation Schemes for Autonomous. Computer-Based Agents," *Technical Report EDRC 18-59-96*, Carnegie Mellon University, Pittsburgh, 1996.

32) Tomek, I., "An Experiment with the Edited Nearest-Neighbor Rule," *IEEE Trans. on Systems, Man, and Cybernetics, 6, 6*, pp. 448-452, 1976.

33) Wilson, D. R., Martinez, T. R., "An integrated instance-based learning algorithm," *Computational Intelligence, 16*, pp. 1-28, 2000.

34) Wilson, D. R., Martinez, T. R., "Reduction Techniques for Instance-based Learning Algorithm," *Machine Learning, 33, 3*, Kluwer Academic Publishers, Boston, pp. 257-286, 2000.

35) Witten, I. H., Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques with JAVA Implementations*, Morgan Kaufmann, San Francisco, 2003.

36) Wroblewski, J., *Adaptacyjne metody klasyfikacji obiektow*, Ph.D. Thesis, University of Warsaw, Warsaw (in Polish), 2001.

37) EL-Manzalawy, Y., Honavar, V., "WLSVM: Integrating LibSVM into Weka Environment," Software available at http://www.cs.iastate.edu/∼yasser/wlsvm, 2005.

38) Zongker, D., Jain, A., "Algorithm for Feature Selection: An Evaluation," in *Proc. of the International Conference on Pattern Recognition, ICPR '96*, pp. 18-22, 1996.

Ireneusz Czarnowski, Ph.D.: He holds B.S. and M.S. in Electronics and Communication Systems from Gdynia Maritime University. In 2004, he obtained Ph.D. in Computer Science from Poznań Technical University. He is presently employed as assistant professor, Department of Information Systems, Faculty of Business Administration, Gdynia Maritime University. His scientific interests include combinatorial optimization, artificial intelligence, data mining and Internet technologies.

Piotr Jędrzejowicz, Ph.D.: He received his M.A., Ph.D. and Dr hab. degrees in operations research from Gdańsk University. Currently, he is a professor of information systems and head of the Information Systems Department, Faculty of Business Administration, Gdynia Maritime University. He is also a Vice-Rector for Research, Gdynia Maritime University. His research interests include decision support systems, computational intelligence, data mining and multi-agent systems.