A New Cluster-based Instance Selection Algorithm

Ireneusz Czarnowski and Piotr Jędrzejowicz

Department of Information Systems, Gdynia Maritime University
Morska 83, 81-225 Gdynia, Poland
{irek,pj}@am.gdynia.pl

Abstract. The main contribution of the paper is proposing and evaluating, through the computational experiment, an agent-based population learning algorithm generating a representative training dataset of the required size. The proposed approach is based on the assumption that prototypes are selected from clusters. Thus, the number of clusters produced has a direct influence on the size of the reduced dataset. Agents within an A-Team execute various local search procedures and cooperate to find-out a solution to the instance reduction problem aiming at obtaining a compact representation of the dataset. Computational experiment has confirmed that the proposed algorithm is competitive to other approaches.

Keywords: data reduction, instance selection, clustering, machine learning, optimization, population learning algorithm, A-Team.

1 Introduction

The paper focuses on data reduction. Data reduction in the supervised machine learning aims at deciding which instances from the training set should be retained for further use during the learning process [11]. Data reduction can be achieved through selecting instances and/or through selecting features.

Data reduction is considered especially useful as a mean to increasing effectiveness of the learning process when the available datasets are large, such as those encountered in data mining, text categorization, financial forecasting, mining of multimedia databases and meteorological, financial, industrial and science repositories, analysing huge string data like genome sequences, Web documents and log data analysis, mining of photos and videos, or information filtering in E-commerce [5, 11, 17, 20]. Finding a small set of representative instances, also called patterns, prototypes or reference vectors, for large datasets can result in producing a classification model superior to one constructed from the whole massive dataset. Besides, such an approach may help to avoid working on the whole original dataset all the time [21]. It is also obvious that removing some instances from the training set reduces time and memory complexity of the learning process [12, 19].

Data reduction is also considered as an important step towards increasing effectiveness of the learning process when the available datasets are large or distributed and when the access to data is limited and costly from the commercial point of view. In the distributed data mining (DDM) a simple approach is to move all of the data

J. O'Shea et al. (Eds.): KES-AMSTA 2011, LNAI 6682, pp. 436–445, 2011.

© Springer-Verlag Berlin Heidelberg 2011

from distributed sites to a central site, merging the data and building a single global learning model. Unfortunately, moving all data into a centralized location can be very time consuming, costly, or may not be feasible due to some restrictions [12, 13, 15].

Selecting the relevant data from distributed locations and then moving only the local patterns can eliminate or reduce the restrictions on a communication bandwidth, reduce the cost of data shipping, and speed up the distributed learning process [8]. The important distributed data mining problem is to establish a reasonable upper bound on the size of the dataset needed for an efficient analysis [18].

The paper deals with instance selection. The main contribution of the paper is proposing and evaluating through computational experiment an agent-based population learning algorithm generating a representative dataset of the required size. The proposed approach is based on the assumption that prototypes are selected from clusters. Thus, the number of clusters produced has direct influence on the size of the reduced dataset. The instance selection method proposed in this paper is an extension of the approach introduced in [6]. In [6] the instance selection was carried in two stages. At the first stage clusters were generated using the similarity coefficient as the criterion for the instance selection procedure. The final number of clusters was obtained by merging the initially generated clusters aiming at minimizing the proximity measures. The merging was repeated several times to obtain the required data compression rate and several merging procedure were considered. In the presented approach the initial clusters are generated in the same way. Next, when the initially generated clusters does not assure the required data compression rate the clusters are merged under the learning process executed by the team of agents.

The goal of the paper is to show through computational experiment that the proposed cluster-based instance selection approach assures the required data compression rate and that it can be competitive to its first version as presented in [6], as well as to several other data reduction algorithms known from the literature. To validate the approach, a number of computational experiment runs have been carried out. Performance of the proposed algorithm has been evaluated using several benchmark datasets from UCI repository [1].

The paper is organized as follows. The agent-based instance selection algorithm and its features are presented in Section 2. Section 3 provides details on the computational experiment setup and discusses its results. Finally, the last section contains conclusions and suggestions for future research.

2 An Approach to the Cluster-based Instance Selection

Since instance selection belongs to the class of computationally difficult combinatorial optimization problems [8], to solve its instances it is proposed to apply one of the population-based metaheuristics known as the population-learning algorithm proposed originally in [10]. In the proposed implementation the optimization and improvement procedures are executed by the set of agents cooperating and exchanging information within an asynchronous team of agents (A-Team).

The section contains an overview of the dedicated agent-based architecture used, including its main features, and gives a detailed description of the proposed agent-based population learning algorithms for the cluster-based instance selection.

2.1 A-Team Concept

The A-Team architecture has been proposed as a problem-solving paradigm that can be easily used to design and implement the proposed population learning algorithm carrying-out the instance reduction tasks. The A-Team concept was originally introduced in [16]. The idea of the A-Team was motivated by several approaches like blackboard systems and evolutionary algorithms, which have proven to be able to help successfully solving some difficult combinatorial optimization problems. Within an A-Team agents achieve an implicit cooperation by sharing a population of solutions to the problem to be solved. Such a population of solutions is an equivalent of the population of individuals known from the evolutionary algorithms.

In the discussed population-based multi-agent approach multiple agents search for the best solution using local search heuristics and population based methods. The best solution 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. All agents can work asynchronously and in parallel. During their work agents cooperate to construct, find and improve solutions which are read from the shared common memory. Their interactions provide for the required adaptation capabilities and for the evolution of the population of potential solutions.

The main functionality of the agent-based population learning approach includes organizing and conducting the process of searching for the best solution. It involves a sequence of the following steps:

- Generation of the initial population of solutions to be stored in the common memory.
- Activation of optimizing agents which apply solution improvement algorithms to solutions drawn from the common memory and store them back after the attempted improvement applying some user defined replacement strategy.
- Continuation of the reading-improving-replacing cycle until a stopping criterion is met. Such a criterion can be defined either or both as a predefined number of iterations or a limiting time period during which optimizing agents do not manage to improve the current best solution. After computation has been stopped the best solution achieved so far is accepted as final.

To implement the agent-based population learning algorithm one has to set and define the following:

- Solution representation format
- Initial population of individuals
- Fitness function
- Improvement procedures
- Replacement strategy implemented for managing the population of individuals.

More information on the population learning algorithm with optimization procedures implemented as agents within an asynchronous team of agents (A-Team) can be found in [2]. In [2] also several its implementations are described.

2.2 Agent-Based Population Learning Algorithm for Instance Selection

This paper proposes an A-Team in which agents execute the improvement procedure and cooperate in a manner described in the preceding subsection with a view to solve instances of the data reduction problem. The approach is called ABIS (Agent-Based Instance Selection).

The basic assumptions behind the proposed agent-based instance selection approach are following:

- Instances are selected from clusters of instances. Clusters are constructed separately from the set of training instances with the identical class label.
- Prototype instances are selected from clusters through the population-based search carried out by the optimizing agents.
- Clusters are induced in two stages. At the first stage the initial clusters are produced using the procedure based on the similarity coefficient. The instances are grouped into clusters according to their similarity coefficient calculated as in [5]. The clusters induced at the first stage contain instances with identical similarity coefficient. The main feature of the discussed clustering procedure is that the number of clusters is determined by the value of the similarity coefficient (see, for example, [5]). The second stage involves merging of clusters obtained at the initial stage. Clusters are merged through the population-based search. The merging is carried-out in case when the number of clusters obtained at the first stage does not assure the required data compression rate. Such case means that the number of clusters obtained at the cluster initialization stage is greater than the upper bound of the number of clusters set by the user. Hence, the solution produced using the similarity coefficient only, is not feasible and must be further improved at the second stage.
- Initially, potential solutions are generated through randomly selecting exactly one single instance from each of the considered clusters (either merged or not merged).
- Each solution from the population is evaluated and the value of its fitness is calculated. The evaluation is carried out by estimating classification accuracy of the classifier, which is constructed using instances (prototypes) indicated by the solution as the training dataset.

A feasible solution is represented by two data structures: a string and a binary square matrix of bits. A string contains numbers (numeric labels) of instances selected as prototypes. A total length of the string is equal to the number of clusters of potential reference instances. The following example explains the notation used. Assume that the number of instances in the original data set is 15, numbered from 1 to 15. Let the considered string be:

$$s = [4,7,12],$$

thus, the number of clusters of potential reference instances equals 3. Instance number 4 from the training dataset has been selected to represent the first cluster as the reference vector. Similarly, instance number 7 has been selected as the reference vector for the second cluster, and instance number 12 as the reference vector for the third one.

A matrix of bits $M = [m_{ij}]_{n \times n}$, where n is the initial number of clusters, denotes whether or not clusters, induced at the cluster initialization stage, have been merged with a view to comply with the constraint on the number of reference vector allowed. The element m_{ij}=1, which lies in the i-th row and the j-th column of the matrix M, denotes that clusters i and j are merged. In addition the discussed matrix has the following properties:

(1) $\sum_{ij} m_{ij} = k$, where k is the upper bound on the allowed number of clusters and hence, number of the reference vectors.

(2) $\forall_i \sum_j m_{ij} = 1$.

(3) $\forall_{i,j:i=j}$ the element m_{ij} of a matrix M is an artificial "missing value".

The following example explains the above notation. Let the number of the required clusters is equal to 3 and the number of clusters induced at the cluster initialization stage is equal to 6. Thus the 6-by-6 matrix should be considered. Let the considered matrix be:

$$M = \begin{bmatrix} - & 1 & 0 & 0 & 0 & 0 \\ 0 & - & 0 & 0 & 0 & 0 \\ 0 & 0 & - & 0 & 1 & 0 \\ 0 & 1 & 0 & - & 0 & 0 \\ 0 & 0 & 0 & 0 & - & 0 \\ 0 & 0 & 0 & 0 & 0 & - \end{bmatrix}$$

where m_{12}=1 denotes that the first cluster is merged with the second, m_{35}=1 denotes that the third cluster is merged with the fifth, and m_{42}=1 denotes that the fourth cluster is merged with the second.

Initially, for each individual in the population of solutions, the corresponding matrix M conforming with properties (1) – (3) is generated randomly. To solve the cluster-based instance selection problem, the following two groups of optimizing agents carrying out different improvement procedures have been implemented:

- The first group includes the improvement procedures for instance selection as it was proposed in [4]. These procedures are: the local search with the tabu list for instance selection and the simple local search. The first procedure - local search with the tabu list for instance selection, modifies a solution by replacing a randomly selected reference instance with some other randomly chosen reference instance thus far not included within the improved solution. The modification takes place providing the replacement move is not on the tabu list. After the modification, the move is placed on the tabu list and remains there for a given number of iterations. The second procedure - simple local search modifies the current solution either by removing the randomly selected reference instance or by adding some other randomly selected reference instance thus far not included within the improved solution.

- The second group consists of the optimizing agents responsible for merging clusters. Agents execute a simple local search procedure. The procedure modifies the current solution changing the composition of clusters through transferring a single reference vector from randomly selected merged cluster into another randomly selected cluster in each iteration. The merging procedure is shown as Algorithm 1.

Algorithm 1. Local search for cluster merging

```
Input: $M_{d:d=1,\dots,m}$ - parts of the solution encoded as square
matrices for clusters containing instances belonging to a
single class. The dimensions of each matrix is $n_d \times n_d$,
where $m$ - denotes the number of the decision classes and
$n_d$ is the initial number of clusters for the class $d$.
Output: $M'_{d:d=1,\dots,m}$ - the improved solutions encoded as a
matrix.
1. Set $d$ by drawing it at random from {1,2,…,$m$};
2. Choose randomly two elements from the matrix $M_d$
   referred to as $m_{ij}$ and $m_{kh}$ (where $i,j,g,h=1,\dots,n_d$) such
   that $m_{ij}=1$ and $m_{gh}=0$;
3. Replace the values between $m_{ij}$ and $m_{gh}$ thus producing
   $M'_d$ satisfying conditions (1) and (2);
4. If ($M'_d$ is better than $M_d$) then goto 6;
5. If (!terminating condition) then goto 1;
6. Return $M'_d$;
```

In each of the above cases the modified solution replaces the current one if it is evaluated as a better one using the classification accuracy as the criterion. If, during the search, an agent successfully improves the solution then it stops and the improved solution is stored in the common memory. Otherwise, agents stop searching for an improvement after having completed the prescribed number of iterations.

The proposed A-Team uses a simple replacement strategy. Each optimizing agent receives a solution drawn at random from the population of solutions (individuals). The solution returned by optimizing agent is merged with the current population replacing the current worst solution.

3 Computational Experiment

To validate the proposed approach it has been decided to carry out the computational experiment. The experiment aimed at answering the following two basic questions:

- Does the ABIS assure appropriate compression rate?
- Does the ABIS perform better than other data reduction algorithms?

To validate the proposed approach several benchmark classification problems have been solved. Datasets for each problem including Cleveland heart disease, credit approval, Wisconsin breast cancer, Ionosphere, Hepatitis, Diabetes and Sonar have been obtained from the UCI Machine Learning Repository [1]. Characteristics of these datasets are shown in Table 1.

Each benchmarking problem has been solved 30 times, and the experiment plan involved 3 repetitions of the 10-cross-validation scheme. The reported values of the quality measure have been averaged over all runs. The quality measure in all cases was the correct classification ratio. In the 10-cross-validation scheme, for each fold, the training dataset was reduced using the proposed approach.

Table 1. Datasets used in the reported experiment

Dataset	Number of instances	Number of attributes	Number of classes	The best reported classification accuracy
Heart	303	13	2	90.0% [7]
Sonar	208	60	2	97.1% [1]
Australian credit (ACredit)	690	15	2	86.9% [1]
German credit (GCredit)	1000	20	2	77.47%[9]
Cancer	699	9	2	97.5% [1]
Ionosphere	351	34	2	94.9% [1]
Hepatitis	155	19	2	87.13%[9]
Diabetes	768	8	2	77.34%[9]

The proposed algorithm has been run three times, with the upper bound on the number of the selected prototypes set to *t%* of the number of instances in the original dataset, where during the experiment *t* was equal to 10, 15 and 20, respectively.

During the experiment population size for each investigated A-Team architecture was set to 20. The process of searching for the best solution has been stopped either after 100 iterations or earlier in case there has been no improvement of the current best solution for one minute of computation. Values of these parameters were set arbitrarily.

The proposed A-Team has been implemented using the middleware environment called JABAT [2], based on JAVA code and built using JADE (Java Agent Development Framework) [3].

Classification accuracy of the classifier obtained using the proposed approach (i.e. using the set of prototypes, found by selecting instances and removing irrelevant attributes) has been compared with:

- Results obtained by machine classification without data reduction, i.e. on full, non-reduced dataset.
- Results obtained using the set of prototypes produced through selection based on the *k*-means clustering (In this case at the first stage the *k*-means clustering has been implemented and next, from thus obtained clusters, the prototypes have been selected using the agent-based population learning algorithm as in [10]).
- Results obtained using the first version of the proposed algorithm (in this case at the second stage the initially generated clusters are merged using the average linkage cluster merging strategy – ALP [6]).

Generalization accuracy has been used as the performance criterion. The learning tool used was the C4.5 algorithm [14]. The experiment results are shown in Table 2. The ranking of the compared approaches is shown in Fig. 1, where horizontal axis represents the mean relative difference between the mean accuracies of the best method and the given method. The above discussed results are averaged over all experiments.

To evaluate the performance of compared algorithms the Friedman's non-parametric test has been used. The test aimed at deciding the following hypotheses:

- H_0 – null hypothesis: instance selection algorithms are statistically equally effective regardless of the kind of the problem.
- H_1 – alternative hypothesis: not all instance selection algorithms are equally effective.

The analysis has been carried out at the significance level of 0.05. The respective value χ^2 statistics with 10 algorithms and 8 instances of the considered problems is 32 and the value of χ^2 distribution is equal to 16.9 for 9 degree of freedom. Analysis of the experiment results shows that for the population of the classification accuracy observations the null hypothesis should be rejected. Thus, not all algorithms are equally effective regardless of the kind of problem which instances are being solved. In Fig. 1 average weights for each instance selection algorithm are shown.

Table 2. Accuracy of the classification results (%)

	Heart	Cancer	ACredit	Sonar	Hep.	GCredit	Ino.	Diab.
C 4.5*	77.89	94.57	84.93	74.04	83.87	70.5	91.45	73.82
ALP (*t*=10%)	84.64	96.30	86.66	78.32	81.34	72.21	88.47	78.3
ALP (*t*=15%)	**87.84**	98.20	88.14	76.62	83.84	74.04	87.13	76.6
ALP (*t*=20%)	86.52	97.62	89.20	74.62	**84.03**	73.1	89.46	77.21
k-means (*t*=10%)	85.67	94.43	88.99	54.34	75.67	68.01	84.54	72.1
k-means (*t*=15%)	87.67	95.09	90.14	59.48	75.32	69.8	83.32	73.7
k-means (*t*=20%)	86.00	96.14	90.29	72.17	76.06	68.4	82.32	74.53
ABIS (*t*=10%)	84.0	97.30	88.55	80.77	79.6	77.2	90.59	80.35
ABIS (*t*=15%)	86.33	**98.86**	**90.69**	80.77	**84.03**	78.1	91.45	**80.61**
ABIS (*t*=20%)	87.26	97.58	89.56	**81.73**	82.04	**78.7**	**92.01**	**80.61**

*- non-reduced data set.

From Fig. 1, one can observe that the best results have been obtained by the ABIS algorithm. The ABIS algorithm assures the required size of the reduced dataset and is competitive to the earlier version – ALP. The worst result have been produced by the *k*-means-based approach. It should be also noted that the cluster-based approach produced very good results as compared with case when the classifier is induced using original, non-reduced dataset.

The experiment results also show that the agent-based search is a suitable tool for solving complex optimization problems including cluster-based instance selection.

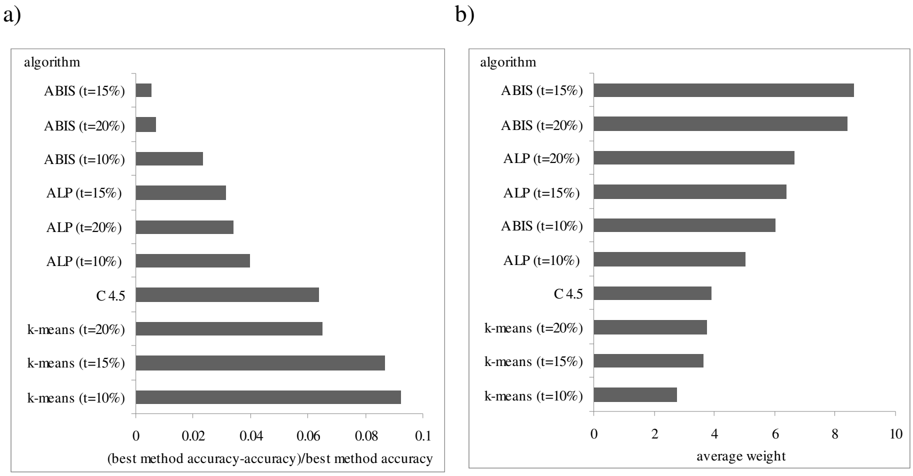

Fig. 1. Ranking of the instance selection algorithms (a) and the average Friedman test weights (b)

4 Conclusions

The paper proposes an approach to instance selection based on cluster integration. The main property of the proposed algorithm is that the instances are selected from clusters, where the number of clusters determines the final size of the reduced dataset. The instances are selected by agent-based population learning algorithm, which role is also to produce the appropriate number of clusters.

The main contribution of the paper is proposing and evaluating through computational experiment a new cluster-based instance selection algorithm. In the reported computational experiment the proposed algorithm outperformed other cluster-based algorithms.

Future research will focus on establishing rules for finding an optimal configuration of the A-Team carrying out the data reduction task.

Acknowledgments. This research has been supported by the Polish Ministry of Science and Higher Education with grant for years 2010-2013.

References

1. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. School of Information and Computer Science. University of California, Irvine (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
2. Barbucha, D., Czarnowski, I., Jędrzejowicz, P., Ratajczak-Ropel, E., Wierzbowska, I.: e-JABAT - An Implementation of the Web-Based A-Team. In: Nguyen, N.T., Jain, I.C. (eds.) Intelligent Agents in the Evolution of Web and Applications. SCI, vol. 167, pp. 57–86. Springer, Heidelberg (2009)

3. Bellifemine, F., Caire, G., Poggi, A., Rimassa, G.: JADE. A White Paper. Exp. 3(3), 6–20 (2003)
4. Czarnowski, I., Jędrzejowicz, P.: An Approach to Data Reduction and Integrated Machine Classification. New Generation Computing 28(1), 21–40 (2010)
5. Czarnowski, I., Jędrzejowicz, P.: An Approach to Instance Reduction in Supervised Learning. In: Coenen, F., Preece, A., Macintosh, A. (eds.) Research and Development in Intelligent Systems XX, pp. 267–282. Springer, London (2004)
6. Czarnowski, I., Jędrzejowicz, P.: Cluster Integration for the Cluster-Based Instance Selection. In: Pan, J.-S., Chen, S.-M., Nguyen, N.T. (eds.) ICCCI 2010. LNCS, vol. 6421, pp. 353–362. Springer, Heidelberg (2010)
7. Datasets used for classification: comparison of results. directory of data sets, `http://www.is.umk.pl/projects/datasets.html` (accessed September 1, 2009)
8. Hamo, Y., Markovitch, S.: The COMPSET Algorithm for Subset Selection. In: Proceedings of The Nineteenth International Joint Conference for Artificial Intelligence, Edinburgh, Scotland, pp. 728–733 (2005)
9. Jędrzejowicz, J., Jędrzejowicz, P.: Cellular GEP-Induced Classifiers. In: Pan, J.-S., Chen, S.-M., Nguyen, N.T. (eds.) ICCCI 2010. LNCS, vol. 6421, pp. 343–352. Springer, Heidelberg (2010)
10. Jędrzejowicz, P.: Social Learning Algorithm as a Tool for Solving Some Difficult Scheduling Problems. Foundation of Computing and Decision Sciences 24, 51–66 (1999)
11. Kim, S.-W., Oommen, B.J.: A Brief Taxonomy and Ranking of Creative Prototype Reduction Schemes. Pattern Analysis Application 6, 232–244 (2003)
12. Klusch, M., Lodi, S., Moro, G.: Agent-Based Distributed Data Mining: The KDEC Scheme. In: Klusch, M., Bergamaschi, S., Edwards, P., Petta, P. (eds.) Intelligent Information Agents. LNCS (LNAI), vol. 2586, pp. 104–122. Springer, Heidelberg (2003)
13. Krishnaswamy, S., Zaslavsky, A., Loke, S.W.: Techniques for Estimating the Computation and Communication Costs of Distributed Data Mining. In: Sloot, P.M.A., Tan, C.J.K., Dongarra, J., Hoekstra, A.G. (eds.) ICCS-ComputSci 2002. LNCS, vol. 2329, pp. 603–612. Springer, Heidelberg (2002)
14. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, SanMateo (1993)
15. Silva, J., Giannella, C., Bhargava, R., Kargupta, H., Klusch, M.: Distributed Data Mining and Agents. Engineering Applications of Artificial Intelligence Journal 18, 791–807 (2005)
16. Talukdar, S., Baerentzen, L., Gove, A., de Souza, P.: Asynchronous Teams: Co-operation Schemes for Autonomous, Computer-Based Agents. Technical Report EDRC 18-59-96, Carnegie Mellon University, Pittsburgh (1996)
17. Uno, T.: Multi-sorting Algorithm for Finding Pairs of Similar Short Substrings from Large-scale String Data. Knowledge and Information Systems (2009); doi: 10.1007/s10115-009-0271-6
18. Vucetic, S., Obradovic, Z.: Performance Controlled Data Reduction for Knowledge Discovery in Distributed Databases. In: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 29-39 (2000)
19. Wilson, D.R., Martinez, T.R.: Reduction Techniques for Instance-based Learning Algorithm. Machine Learning 33(3), 257–286 (2000)
20. Yu, K., Xiaowei, X., Ester, M., Kriegel, H.-P.: Feature Weighting and Instance Selection for Collaborative Filtering: An Information-Theoretic Approach. Knowledge and Information Systems 5(2), 201–224 (2004)
21. Zhu, X., Wu, X.: Scalable Representative Instance Selection and Ranking. In: IEEE Proceedings of the 18th International Conference on Pattern Recognition, vol. 3, pp. 352–355 (2006)