# Ensembles of classifiers based on approximate reducts

Jakub Wróblewski

Polish-Japanese Institute of Information Technology
and
Institute of Mathematics, Warsaw University
Banacha 2, 02-097 Warsaw, Poland
e-mail: jakubw@mimuw.edu.pl
http://alfa.mimuw.edu.pl/~jakubw/

**Abstract.** A problem of improving rough set based expert systems by modifying a notion of reduct is discussed. A notion of approximate reduct is introduced, as well as some proposals of quality measure for such a reduct. A complete classifying system based on approximate reducts is presented and discussed. It is proved that a problem of finding optimal set of classifying agents based on approximate reducts is NP-hard; a genetic algorithm is used to find the suboptimal set. Experimental results show, that the classifying system is effective and relatively fast.

## 1 Introduction

Rough set expert systems base on the notion of *reduct* [11], [12], a minimal subset of attributes which is sufficient to discern between objects with different decision values. A set of short reducts can be used to generate rules [2]. A problem of short reducts generation is NP-hard, but an approximate algorithm (like the genetic one described in [13], [8] and implemented successfully – see [10]) can be used to obtain reducts in reasonable time. On the other hand, rules generated basing on reducts are often too specific and cannot classify new objects. Another types of reducts have been considered to improve efficiency on new objects (e.g. *dynamic reducts* – see [3], or reducts optimized by number of generated rules – see [16]). One of the methods is to calculate reducts basing on a single object [16]; results are good, but calculation time may be long, even when parallel algorithm is used [15].

A notion of *approximate reduct* and *classifying agent* is introduced in the next sections. A method for fast approximate reducts generation and evolutionary process (based on genetic algorithm) of expert system tuning is presented.

## 2 Approximate reducts

Let $\mathbb{A} = (U, A \cup \{d\})$ be an *information system* (see [12]), where $U$ – set of objects, $A$ – set of attributes, $d$ – decision.

**Definition 1.** *An* ***approximate reduct*** *with respect to a* ***reduct quality measure*** $Q : 2^A \to \mathbb{R}$ *is a subset* $R \subseteq A$ *such that:*
*a)* $\forall\ R' \subset R,\ R' \neq R,\ \ Q(R') < Q(R)$
*b)* $\forall\ R'' \supset R,\ R'' \neq R,\ Q(R'') \leq Q(R)$
*i.e.* $R$ *is a local maximum of measure function* $Q$.

Every approximate reduct $R$ (and, in general, every subset of attributes) defines an indiscernibility relation [11] on a set of objects, i.e. two objects are in relation iff values of attributes included in $R$ are equal. Hence, $R$ defines a partition of set of objects $U$. Every abstract class of this relation generates one (generalized) decision rule:

$$r_i = (a_{i_1} = v_{i_1} \wedge ... \wedge a_{i_{|R|}} = v_{i_{|R|}} \Rightarrow d = (\partial_{i_1}, ..., \partial_{i_n}))$$

where $A = \{a_1, ..., a_{|A|}\}$ is a set of attributes, $n$ is a number of decision classes, $\{\partial_{i_1}, ..., \partial_{i_n}\}$ is a probability distribution of decision value, based on the distribution on the abstract class ($\partial_i \in [0, 1]$).

When one have to take into account only one decision value rather than the whole distribution, the maximal value can be used:

$$r_i = (a_{i_1} = v_{i_1} \wedge ... \wedge a_{i_{|R|}} = v_{i_{|R|}} \Rightarrow d = d_j)$$

where $j$ is such that $\partial_{i_j} = max(\partial_{i_1}, ..., \partial_{i_n})$.

To increase the quality of expert system, some rules should be removed from the final rule set. Namely, for a given constant $\alpha \in [0,1]$, one should remove rules for which $\partial_{i_j} < \alpha$. Unfortunately, optimal parameter $\alpha$ can be found only by experiments; for some testing databases the best results were obtained for $\alpha = 0$, whereas in another case for $\alpha = 0.9$.

The quality measure should take into account two aspects: a degree the subset is a reduct and its ability to generate good rule sets. Consider the following quality measure:

$Q(R) = |A| - |R|$ , when R is a reduct; $Q(R) = 0$ otherwise.

It is easy to see, that for the above quality measure, Definition 1 is equivalent of a definition of reduct; moreover, the quality values of short reducts are higher than those of long ones. Unfortunately, the systems based on classical short reducts are often too specific (there are many not recognized objects).

To obtain more effective set of rules we use another quality measure, originally used to evaluate new features in databases [17]:

**Definition 2.** ***Predictive quality measure*** *of subsets $R$ of attributes is defined as:*

$$Q(R) = \sqrt[n]{\prod_{i=1}^{n} \frac{P(\{r_1 ... r_k\}, \mathbb{A}, i)}{P(\{r_1 ... r_k\}, \mathbb{A}, i) + N(\{r_1 ... r_k\}, \mathbb{A}, i)}} \times P_{cov} \tag{1}$$

$$P_{cov} = 1 - \prod_{i=1}^{k} (1 - \frac{n(r_i) - 1}{|U| - 1}) \tag{2}$$

*where $\mathbb{A} = (U, A \cup \{d\})$ – training decision table, $k$ – number of rules generated by $R$, $n(r_i)$ – number of training objects classified properly by rule $r_i$, $P(\{r_1 ... r_k\}, \mathbb{A}, i)$ – number of properly classified objects belonging to $i$-th decision class, $N(\{r_1 ... r_k\}, \mathbb{A}, i)$ – number of objects belonging to $i$-th decision class but classified to another one, $n$ – number of decision classes.*

Experiments show, that $Q$ provides good estimation of final classification algorithm quality on a real data. Approximate reducts optimized by $Q$ have better forecasting capabilities than e.g. short ones. One uses short reducts because rules based on less attributes should be more general than others (minimum description length principle). On the other hand, predictive quality measure estimates an effectiveness of expert system on unknown testing data table, so reducts (and respective sets of rules) are optimized "more directly" [6]. From now on, by *approximate reduct* we will mean a reduct optimized by predictive quality measure.

A connection between approximate reducts and classical ones is given by the following fact:

**Theorem 1.** *If $R$ is a reduct (in classical sense – see [11], [12]), then $R$ satisfies condition b) in approximate reduct definition (with respect to predictive quality measure).*

We have used a simple heuristics to generate approximate reducts. First, a random permutation $\sigma$ of attributes is generated. Then, according to this permutation, attributes are added to a subset $R$ and its quality is calculated. Typically, quality value is low for small subsets, and increases when the next attribute is added. When quality starts to decrease, next phase begins. Each attribute from $R$ is replaced one by one and a quality measure is calculated – an attribute causing the highest quality increase is replaced from $R$. The algorithm stops when local optimum is achieved (the result is not always an approximate reduct; the algorithm just approximates it). If two subsets have the same quality measure, the shorter one (by means of number of attributes) is taken.

Note, that ordering $\sigma$ of attributes together with $R$ generates an ordering $\tau$ of set of objects $U$: objects are sorted by attributes' values. Two additional techniques was used to improve final classification quality. First, a generalization method for rules was used: every three adjacent (by means of objects ordering $\tau$) rules was analyzed and some of them was joined (generalized) if system quality was higher after this operation. Second, a method of unrecognized objects classification was introduced: in some cases when a new object does not math any rule but we have to classify it anyway, we can use *upper approximation* [11] of rule set. In this case we find the closest rules (by means of objects ordering $\tau$) and use them to determine a decision value. Hence, the complete classifying system based on an approximate reduct $R$ should contain not only a set of rules generated by $R$, but also the ordering $\tau$.

**Definition 3.** *By **classifying agent based on approximate reduct** we will denote a triple $(Rul, \mathbb{A}, \tau)$, where $Rul$ is a set of decision rules generated by $R$, based on values of attributes of a training information system $\mathbb{A}$, $\tau$ – a permutation of objects of $\mathbb{A}$.*

## 3 Optimal set of classifying agents

Other reduct-based systems [10] generate several well-optimized reducts (e.g. using genetic algorithm) and use all of them to create rule set. In some approaches [1] the rule set is then filtered (e.g. by genetic algorithm). Our strategy is different: generate many reducts (classifying agents) using fast approximation heuristics (maybe these reducts will not be optimal), then construct a classification system by selecting optimal subset of them. If some agents will be worse than others, they simply will not be used in final system. On the other hand, even very poor (when evaluated separately) agent may become valuable e.g. because it can classify some objects which are hard to recognize by other agents in a team.

We will optimize a set of agents due to classification results on a testing set. If two sets of agents perform equally, we will select one with the less agents.

**Theorem 2.** *Problem of selecting optimal subset of classifying agents based on approximate reducts is NP-hard. (We will take into account neither voting technique nor rule generalization).*

**Proof:**

We will show, that any minimal row covering problem for binary matrices (MATRIXCOVER, see [4]) can be solved by selecting optimal subset of classifying agents in a case of special data table. Let $\mathbf{B} = \{b_{ij}\}$ be a binary matrix of size $n \times m$. Our goal is to find a minimal set of columns such that in every row there is at least one "1" in a selected column.

A special information system $\mathbb{A} = (U, A \cup \{d\})$ is constructed in the following way. Every row in matrix $\mathbf{B}$ corresponds to a pair of objects in $U$; an additional set of $2m$ objects is used. Every column in matrix $\mathbf{B}$ corresponds to one attribute in $\mathbb{A}$. So $|A| = n$, $|U| = 2m + 2m$. Attributes and decision values for $\mathbb{A}$, for the first $2m$ objects:

$a_i(u_{2j-1}) = b_{ij}(j+1)$,

$a_i(u_{2j}) = b_{ij}j + 1$,

$d(u_{2j}) = d(u_{2j-1}) = j$, where $j = 1...m$.

Attributes and decision values for $\mathbb{A}$, for the next $2m$ objects:

$a_i(u_{2m+2j-1}) = 0$,

$a_i(u_{2m+2j}) = 1$,

$d(u_{2m+2j}) = d(u_{2m+2j-1}) = j$, where $j = 1...m$ (see example – Table 1.).

| | | | |
|---|---|---|---|
| **1** | 0 | **1** | 0 |
| **1** | 0 | 0 | **1** |
| 0 | 0 | **1** | 0 |
| 0 | 0 | **1** | **1** |

$\longrightarrow$

| $a_1$ | $a_2$ | $a_3$ | $a_4$ | $d$ |
|---|---|---|---|---|
| **2** | 0 | **2** | 0 | 1 |
| **2** | 1 | **2** | 1 | 1 |
| **3** | 0 | 0 | **3** | 2 |
| **3** | 1 | 1 | **3** | 2 |
| 0 | 0 | **4** | 0 | 3 |
| 1 | 1 | **4** | 1 | 3 |
| 0 | 0 | **5** | **5** | 4 |
| 1 | 1 | **5** | **5** | 4 |
| 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| | ⋮ | | | ⋮ |
| 0 | 0 | 0 | 0 | 4 |
| 1 | 1 | 1 | 1 | 4 |

Table 1. Matrix $\mathbf{B}$ and information system $\mathbb{A}$.

Let us produce a set of approximate reducts using predictive measure for $\alpha$ high enouhh, e.g. $\alpha = 0.9$. Let $R_k = \{a_k\}$. Consider $R_1$ in the example presented above – it generates the following rules: $r_1 = (a_1 = 2 \Rightarrow d = 1)$, $r_2 = (a_1 = 3 \Rightarrow d = 2)$, $r_3 = (a_1 = 0 \Rightarrow d = \{1/6, 1/6, 2/6, 2/6\})$, $r_4 = (a_1 = 1 \Rightarrow d = \{1/6, 1/6, 2/6, 2/6\})$. Rules $r_3$ and $r_4$ will be removed because a support of the most supported decision class is lower than $\alpha$. This is a general rule for $R_k$ family: only rules corresponding to $b_{ij} = 1$ will be taken into account; any rule with attribute value 0 or 1 will be filtered out due to a special form of the second part of decision table.

Let us calculate a quality measure for $R_k$. Let $x$ be a number of ones in $k$-th column of $\mathbf{B}$. Note that all (not filtered) rules generated by $R_k$ are accurate and covers 2 objects. Then $Q(R_k) = 1 \times (1 - (1 - \frac{1}{|U|-1})^x)$, which is monotonous with respect to $x$. Note that when we consider larger sets, the quality measure is never higher. Consider e.g. $R = \{a_1, a_2\}$. In this case rule $r_1$ generated by $R_1$ will be divided into two other: $r_1^1 = (a_1 = 2 \wedge a_2 = 0 \Rightarrow d = 1)$ and

$r_1^2 = (a_1 = 2 \wedge a_2 = 1 \Rightarrow d = 1)$; quality of $R$ will be 0 because of low support of these new rules. In general, quality of a subset containing two attributes will be always less than quality of singletons, unless the corresponding columns in $\mathbf{B}$ are equal. But, in this case, we will rather prefer a set containing one attribute. Thus, the set of approximate reducts for this information system contains sets $R_1$,..., $R_n$ only.

Now we will prove, that optimal set of reducts (agents) corresponds to the optimal set of columns. We will use $m$ objects from $\mathbb{A}$ as a test table, by taking only even indexes from the first $2m$ objects. Note that using $k$-th agent we will classify correctly any object $j$ such that $b_{kj} = 1$. Thus, the optimal (complete) classification corresponds to the columns which cover $\mathbf{B}$. On the other hand, number of agents used in a system is equal to a number of columns covering $\mathbf{B}$. Optimal set of agents has minimal cardinality and corresponds to a minimal covering of $\mathbf{B}$.

It was shown, that for any binary matrix $\mathbf{B}$ we can construct an information system and a set of classifying agents (approximate reducts) such that finding an optimal set of agents gives a minimal column covering of $\mathbf{B}$. As MATRIXCOVER is NP-hard, the problem of finding optimal set of agents is NP-hard too. □

Due to its NP-hardness, the problem of optimal set of agents selection cannot be solved exactly in reasonable time. A genetic algorithm [5] were used in our experiments to choose the best agents in approximate way. Chromosomes (sets of agents) were evaluated by their effectiveness on a separate testing data set (selected randomly at the beginning of training process). Such a classification process is often very time-consuming (in general, time is proportional to [number of rules]×[number of testing objects]), but in our case, because of reduct-based rule generation, a fast $O(|A|^2 \times |U| \log |U|)$ testing algorithm can be used.

## 4 Results of experiments

Several well known benchmark databases published in StatLog project [7] were used in experiments. Table 2. presents results, including data size, calculation time (Celeron 400 MHz) and average error rate. Parameters of algorithm: 50 reducts have been found, then the reducts were filtered by genetic algorithm (population: 60, evolution steps: about 300) and used in classification algorithm. Typically only 10-20 reducts out of initial number of 50 were used in the final classification algorithm.

| Data | Size (obj.×attrib.) | Time | Error |
|---|---|---|---|
| sat image | 4435 × 37 | 223.0 | 0.131 |
| letter | 15000 × 17 | 1310.0 | 0.091 |
| diabetes | 768 × 9 | 5.5 | 0.267 |
| breast cancer | 286 × 10 | 2.3 | 0.268 |
| primary tumor | 339 × 18 | 25.5 | 0.617 |
| Australian credit | 690 × 15 | 5.9 | 0.140 |
| vehicle | 846 × 19 | 16.1 | 0.319 |
| DNA splices | 2000 × 181 | 47.0 | 0.061 |

Table 2. Experimental results.

Calculation time (in seconds) includes classificator generation and testing on test table. If database does not contain separate test table, cross-validation method is used. Results are in many cases better than those obtained by classical methods (C4.5, k-NN, neural nets; see [7]). This is worth noting that error rates presented in table are average of 10 experiments; in several experiments (due to their nondeterministic nature) results were significantly better – e.g. 0.128 for "sat image" data, 0.556 for "primary tumor" data, 0.127 for "Australian credit" data.

Relatively long calculation time for "letter" database is concerned with high number (26) of decision classes rather than with number of objects. We did not use databases smaller than about 200 objects because of low stability of results: about 25% of training objects are used as internal testing sample; when this sample is too small, genetic algorithm cannot optimize the set of agents well enough.

## 5 Conclusions

A classification system based on approximate reducts was presented. As shown in Section 3, problem of the optimal classifying agents selection is NP-hard, so the only way to construct such a set effectively is to use approximate adaptive technique (e.g. genetic algorithm based on system performance on testing data). The system described above proved to be effective and relatively fast on several benchmark data sets.

The paper does not address a problem of voting technique. Experiments with various voting techniques as well as on incorporating voting parameters into genetic algorithm are in progress.

### Acknowledgements

"Primary tumor" and "breast cancer" domains were obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. Thanks go to M. Zwitter and M. Soklic for providing the data.

## References


1. Ågontes T., Komorowski J., Løken T., 1999. *Taming Large Rule Models in Rough Set Approaches.* Proc. of PKDD'99, Prague, Czech Republic. Springer-Verlag (LNAI 1704), Berlin Heidelberg 1999, pp. 193–203.
2. Bazan J., Skowron A., Synak P., 1994. *Dynamic reducts as a tool for extracting laws from decision tables*, Proc. of the Symp. on Methodologies for Intelligent Systems, Charlotte, NC, October 16-19, 1994, Lecture Notes in Artificial Intelligence 869, Springer-Verlag, Berlin 1994, 346-355, also in: ICS Research Report 43/94, Warsaw University of Technology.
3. Bazan J., 1998. *A Comparison of Dynamic and non-Dynamic Rough Set Methods for Extracting Laws from Decision Tables.* In: L. Polkowski, A. Skowron (eds.). Rough Sets in Knowledge Discovery. Physica Verlag, 1998.
4. Garey M. R., Johnson D. S., 1979. *Computers and Intractability, a Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, San Francisco.
5. Goldberg D.E., 1989. *GA in Search, Optimisation, and Machine Learning.* Addison-Wesley.
6. Liu H., Motoda H., 1998. *Feature selection for knowledge discovery and data mining.* Kluwer, Dordrecht.
7. Michie D., Spiegelhalter D.J., Taylor C.C. (ed.), 1994. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood Limited, 1994. Data avaliable at: http://www.ics.uci.edu/~mlearn/MLRepository.html.
8. Nguyen S. H., Skowron A., Synak P., Wróblewski J., 1997. *Knowledge Discovery in Databases: Rough Set Approach.* Proc. of The Seventh International Fuzzy Systems Association World Congress, vol. II, pp. 204-209, IFSA97, Prague, Czech Republic.
9. Nguyen H. S., Nguyen S. H., 1998. *Discretization Methods in Data Mining.* In: L. Polkowski, A. Skowron (eds.). Rough Sets in Knowledge Discovery. Physica Verlag, 1998.
10. Øhrn A., Komorowski J., 1997. *Rosetta – A rough set toolkit for analysis of data.* Proc. of Third International Join Conference on Information Sciences (JCIS97), Durham, NC, USA, March 1 - 5, 3 (1997), pp. 403-407.
11. Pawlak Z., 1991. *Rough sets: Theoretical aspects of reasoning about data.* Kluwer, Dordrecht 1991.
12. Skowron A., Rauszer C., 1992. *The Discernibility Matrices and Functions in Information Systems.* In: R. Słowiński (ed.): Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory. Kluwer, Dordrecht 1992, pp: 331 - 362.
13. Wróblewski J., 1995. *Finding minimal reducts using genetic algorithms.* Proc. of the Second Annual Join Conference on Information Sciences, pp. 186-189, September 28-October 1, 1995, Wrightsville Beach, NC. Also in: ICS Research report 16/95, Warsaw University of Technology.
14. Wróblewski J., 1996. *Theoretical Foundations of Order-Based Genetic Algorithms.* Fundamenta Informaticae, vol. 28 (3, 4), pp: 423-430. IOS Press, 1996.
15. Wróblewski J., 1998: *A Parallel Algorithm for Knowledge Discovery System.* Proc. of PARELEC'98, Bialystok, Poland. The Press Syndicate of the Technical University of Bialystok 1998, pp. 228–230.
16. Wróblewski J., 1998. *Covering with reducts – a fast algorithm for rule generation.* Proc. of RSCTC'98, Warsaw, Poland. Springer-Verlag (LNAI 1424), Berlin Heidelberg 1998, pp. 402–407.
17. Wróblewski J., 2000. *Analyzing relational databases using rough set based methods.* Accepted to IPMU 2000, Madrid.