On Combined Classifiers, Rule Induction and Rough Sets

Jerzy Stefanowski

Institute of Computing Science
Poznań University of Technology,
60-965 Poznań, ul.Piotrowo 2, Poland
`Jerzy.Stefanowski@cs.put.poznan.pl`

Abstract. Problems of using elements of rough sets theory and rule induction to create efficient classifiers are discussed. In the last decade many researches attempted to increase a classification accuracy by combining several classifiers into integrated systems. The main aim of this paper is to summarize the author's own experience with applying one of his rule induction algorithm, called MODLEM, in the framework of different combined classifiers, namely, the bagging, n^2–classifier and the combiner aggregation. We also discuss how rough approximations are applied in rule induction. The results of carried out experiments have shown that the MODLEM algorithm can be efficiently used within the framework of considered combined classifiers.

1 Introduction

Rough sets theory has been introduced by Professor Zdzisław Pawlak to analyse granular information [25, 26]. It is based on an observation that given information about objects described by attributes, a basic relation between objects could be established. In the original Pawlak's proposal [25] objects described by the same attribute values are considered to be *indiscernible*. Due to limitations of available information, its natural granulation or vagueness of a representation language some elementary classes of this relation may be *inconsistent*, i.e. objects having the same descriptions are assigned to different categories. As a consequence of the above inconsistency it is not possible, in general, to precisely specify a set of objects in terms of elementary sets of indiscernible objects. Therefore, Professor Zdzisław Pawlak introduced the concept of the *rough set* which is a set characterized by a pair of precise concepts – *lower* and *upper approximations* constructed from elementary sets of objects.

This quite simple, but smart, idea is the essence of the Pawlak's theory. It is a starting point to other problems, see e.g. [27, 20, 26, 9]. In particular many research efforts have concerned *classification* of objects represented in data tables. Studying relationships between elementary sets and categories of objects (in other terms, target concepts or decision classes in the data table) leads to, e.g., evaluating dependency between attributes and objects classification, determining the level of this dependency, calculating importance of attributes for objects classification, reducing the set of attributes or generating decision rules from data. It is also said that the aim is to synthesize reduced, approximate models of concepts from data [20]. The *transparency* and *explainability* of such models to human is an important property. Up to now rough sets based approaches were applied to many practical problems in different domains – see, e.g., their list presented in [20].

Besides "classical" rough sets, based on the indiscernibility relation, several generalizations have been introduced. Such data properties as, e.g., imprecise attribute values, incompleteness, preference orders, are handled by means of tolerance, similarity, fuzzy valued or dominance relations [9, 20, 37].

Looking into the previous research on rough sets theory and its applications, we could distinguish two main perspectives: *descriptive* and *predictive* ones.

The descriptive perspective includes extraction information patterns or regularities, which characterize some properties hidden in available data. Such patterns could facilitate understanding dependencies between data elements, explaining circumstances of previous decisions and generally gain insight into the structure of the acquired knowledge. In this context presentation of results in a human readable form allowing an interpretation is a crucial issue.

The other perspective concerns *predicting* unknown values of some attributes on the basis of an analysis of previous examples. In particular, it is a prediction of classes for new object. In this context rough sets and rules are used to construct a classifier that has to classify new objects. So, the main evaluation criterion is a *predictive classification accuracy*. Let us remind that the predictive classification has been intensively studied since many decades in such fields as machine learning, statistical learning, pattern recognition. Several efficient methods for creating classifiers have been introduced; for their review see, e.g., [16, 19, 23]. These classifiers are often constructed with using a search strategy optimizing criteria strongly related to predictive performance (which is not directly present in the original rough sets theory formulation). Requirements concerning interpretability are often neglected in favor of producing complex transformations of input data – an example is an idea of support vector machines.

Although in both perspectives we could use the same knowledge representation – rules, since motivation and objectives are distinct, algorithmic strategies as well as criteria for evaluating a set of rules are quite different. For instance, the prediction perspective directs an interest to classification ability of the complete rules, while in the descriptive perspective each rule is treated individually as a possible representative of an 'interesting' pattern evaluated by measures as confidence, support or coverage - for a more exhaustive discussion see, e.g., [42].

In my opinion, basic concepts of the rough sets theory have been rather considered in the way similar to a descriptive analysis of data tables. Nevertheless, several authors have developed their original approaches to construct decision rules from rough approximations of decision classes which joined together with classification strategies led to good classifiers, see e.g. [1, 11, 20, 34, 37]. It seems to me that many authors moved their interest to this direction in the 90's because of at least two reasons: (1) a research interest to verify whether knowledge derived from "closed world" of the data table could be efficiently applied to new objects coming from the "open world" – not seen in the analysed data table; (2) as a result of working with real life applications.

Let us also notice that the majority of research has been focused on developing *single classifiers* – i.e. based on the single set of rules. However, both empirical observations and theoretical works confirm that one cannot expect to find one single approach leading to the best results on overall problems [6]. Each learning algorithm has its *own area of superiority* and it may outperform others for a specific subset of classification problems while being worse for others. In the last decade many researches attempted to increase classification accuracy by combining several single classifiers into an *integrated system*. These are sets of learned classifiers, whose individual predictions are combined to produce the final decision. Such systems are known under names: *multiple classifiers, ensembles* or *committees* [6, 45]. Experimental evaluations shown that these classifiers are quite effective techniques for improving classification accuracy. Such classifiers can be constructed in many ways, e.g., by changing the distributions of examples in the learning set, manipulating the input features, using different learning algorithms to the same data, see e.g. reviews [6, 45, 36]. Construction of integrated classifiers has also attracted the interest of some rough sets researchers, see e.g. [2, 8, 24]. The author and his co-operators have also carried out research, first on developing various rule induction algorithms and classification strategies (a review is given in [37]), and then on multiple classifiers [18, 36, 38, 40, 41].

The main aim of this paper is to summarize the author's experience with applying one of his rule induction algorithm, called MODLEM [35], in the framework of different multiple classifiers: the popular bagging approach [4], the n^2-classifier [18] – a specialized approach to solve multiple class learning problems, and the combiner approach to merge predictions of heterogeneous classifiers including also MODLEM [5]. The second aim is to briefly discuss the MODLEM rule induction algorithm and its experimental evaluation.

This paper is organized as follows. In the next section we shortly discuss rule induction using the rough sets theory. Section 3 is devoted to the MODLEM algorithm. In section 4 we briefly present different approaches to construct multiple classifiers. Then, in the successive three sections we summarize the experience of using rule classifiers induced by MODLEM in the framework of three different multiple classifiers. Conclusions are grouped in section 8.

2 Rules Generation and Rough Sets

2.1 Notation

Let us assume that objects – learning examples for rule generation – are represented in *decision table* $DT = (U, A \cup \{d\})$, where U is a set of objects, A is a set of *condition attributes* describing objects. The set V_a is a domain of a. Let $f_a(x)$ denotes the value of attribute $a \in A$ taken by $x \in U$; $d \notin A$ is a decision attribute that partitions examples into a set of decision classes $\{K_j : j = 1, \ldots, k\}$.

The *indiscernibility relation* is the basis of Pawlak's concept of the rough set theory. It is associated with every non-empty subset of attributes $C \subseteq A$ and $\forall x, y \in U$ is defined as $xI_C y \Leftrightarrow \{(x,y) \in U \times U \;\; f_a(x) = f_a(y) \;\; \forall a \in C\; \}$.

The family of all equivalence classes of relation $I(C)$ is denoted by $U/I(C)$. These classes are called *elementary sets*. An elementary equivalence class containing element x is denoted by $I_C(x)$.

If $C \subseteq A$ is a subset of attributes and $X \subseteq U$ is a subset of objects then the sets: $\{x \in U : I_C(x) \subseteq X\}$, $\{x \in U : I_C(x) \cap X \neq \emptyset\}$ are called *C-lower* and *C-upper* approximations of X, denoted by $\underline{C}X$ and $\overline{C}X$, respectively. The set $BN_C(X) = \underline{C}X - \overline{C}X$ is called the *C-boundary* of X.

A decision rule r describing class K_j is represented in the following form:

$$\textit{if } P \textit{ then } Q,$$

where $P = w_1 \wedge w_2 \wedge \ldots w_p$ is a condition part of the rule and Q is decision part of the rule indicating that example satisfying P should be assigned to class K_j. The elementary condition of the rule r is defined as $(a_i(x) \; rel \; v_{a_i})$, where rel is a relational operator from the set $\{=, <, \leq, >, \geq\}$ and v_{a_i} is a constant being a value of attribute a_i.

Let us present some definitions of basic rule properties. $[P]$ is a *cover* of the condition part of rule r in DT, i.e. it is a set of examples, which description satisfy elementary conditions in P. Let B be a set of examples belonging to decision concept (class K_j or its appropriate rough approximation in case of inconsistencies). The rule r is discriminant if it distinguishes positive examples of B from its negative examples, i.e. $[P] = \bigcap [w_i] \subseteq B$. P should be a minimal conjunction of elementary conditions satisfying this requirement. The set of decision rules R completely describes examples of class K_j, if each example is covered by at least one decision rules.

Discriminant rules are typically considered in the rough sets literature. However, we can also construct *partially discriminant* rules that besides positive examples could cover a limited number of negative ones. Such rules are characterized by the *accuracy* measure being a ratio covered positive examples to all examples covered by the rule, i.e. $[P \cap B]/[P]$.

2.2 Rule generation

If decision tables contain inconsistent examples, decision rules could be generated from rough approximations of decision classes. This special way of treating inconsistencies in the input data is the main point where the concept of the rough sets theory is used in the rules induction phase. As a consequence of using the approximations, induced decision rules are categorized into *certain* (discriminant in the sense of the previous definition) and *possible* ones, depending on the used lower and upper approximations, respectively.

Moreover, let us mention other rough sets approaches that use information on class distribution inside boundary and assign to lower approximation these inconsistent elementary sets where the majority of examples belong to the given class. This is handled in the *Variable Precision Model* introduced by Ziarko [47] or *Variable Consistency Model* proposed by Greco *et al.* [10] – both are a subject of many extensions, see e.g. [31]. Rules induced from such variable lower approximations are not certain but partly discriminant ones.

A number of various algorithms have been already proposed to induce decision rules – for some reviews see e.g. [1, 11, 14, 20, 28, 34, 37]. In fact, there is no unique "rough set approach" to rule induction as elements of rough sets can be used on different stages of the process of induction and data pre-processing. In general, we can distinguish approaches producing *minimal set of rules* (i.e. covering input objects using the minimum number of necessary rules) and approaches generating more extensive rule sets.

A good example for the first category is LEM2, MODLEM and similar algorithms [11, 35]. The second approaches are nicely exemplified by *Boolean reasoning* [28, 29, 1]. There are also specific algorithms inducing the set of decision rules which satisfy user's requirements given a priori, e.g. the threshold value for a minimum number of examples covered by a rule or its accuracy. An example of such algorithms is Explore described in [42]. Let us comment that this algorithm could be further extended to handle imbalanced data (i.e. data set where one class – being particularly important – is under-represented comparing to cardinalities of other classes), see e.g. studies in [15, 43].

3 Exemplary Rule Classifier

In our study we will use the algorithm, called MODLEM, introduced by Stefanowski in [35]. We have chosen it because of several reasons. First of all, the union of rules induced by this algorithm with a classification strategy proved to provide efficient single classifiers, [14, 41, 37]. Next, it is designed to handle various data properties not included in the classical rough sets approach, as e.g. numerical attributes without its pre-discretization. Finally, it produces the set of rules with reasonable computational costs – what is important property for using it as a component inside combined classifiers.

3.1 MODLEM algorithm

The general schema of the MODLEM algorithm is briefly presented below. More detailed description could be found in [14, 35, 37]. This algorithm is based on the idea of a *sequential covering* and it generates a *minimal set* of decision rules for every decision concept (decision class or its rough approximation in case of inconsistent examples). Such a minimal set of rules (also called *local covering* [11]) attempts to cover all *positive examples* of the given decision concept, further denoted as B, and not to cover any *negative examples* (i.e. $U \setminus B$). The main procedure for rule induction scheme starts from creating a first rule by choosing sequentially the 'best' elementary conditions according to chosen criteria (see the function *Find best condition*). When the rule is stored, all learning positive examples that match this rule are removed from consideration. The process is repeated while some positive examples of the decision concept remain still uncovered. Then, the procedure is sequentially repeated for each set of examples from a succeeding decision concept.

In the MODLEM algorithm numerical attributes are handled during rule induction while elementary conditions of rules are created. These conditions are represented as either $(a < v_a)$ or $(a \geq v_a)$, where a denotes an attribute and v_a is its value. If the same attribute is chosen twice while building a single rule, one may also obtain the condition $(a = [v_1, v_2))$ that results from an intersection of two conditions $(a < v_2)$ and $(a \geq v_1)$ such that $v_1 < v_2$. For nominal attributes, these conditions are $(a = v_a)$ or could be extended to the set of values.

```
Procedure MODLEM
(input $B$ - a set of positive examples from a given decision concept;
    criterion - an evaluation measure;
output $\mathcal{T}$ – single local covering of $B$, treated here as rule condition parts)
begin
    $G := B$; {A temporary set of rules covered by generated rules}
    $\mathcal{T} := \emptyset$;
    while $G \neq \emptyset$ do {look for rules until some examples remain uncovered}
    begin
        $T := \emptyset$; {a candidate for a rule condition part}
        $S := U$; {a set of objects currently covered by $T$}
        while $(T = \emptyset)$ or (not($[T] \subseteq B$)) do {stop condition for accepting a rule}
        begin
            $t := \emptyset$; {a candidate for an elementary condition}
            for each attribute $q \in C$ do {looking for the best elementary condition}
            begin
                new_t :=Find_best_condition($q, S$);
                if Better(new_t, $t$, criterion) then $t :=$ new_t;
                {evaluate if a new condition is better than previous one
                according to the chosen evaluation measure}
            end;
            $T := T \cup \{t\}$; {add the best condition to the candidate rule}
            $S := S \cap [t]$; {focus on examples covered by the candidate}
        end; { while not($[T] \subseteq B$ }
        for each elementary condition $t \in T$ do
            if $[T - t] \subseteq B$ then $T := T - \{t\}$; {test a rule minimality}
        $\mathcal{T} := \mathcal{T} \cup \{T\}$; {store a rule}
        $G := B - \bigcup_{T \in \mathcal{T}} [T]$ ; {remove already covered examples}
    end; { while $G \neq \emptyset$ }
    for each $T \in \mathcal{T}$ do
        if $\bigcup_{T' \in \mathcal{T} - T} [T'] = B$ then $\mathcal{T} := \mathcal{T} - T$ {test minimality of the rule set}
end {procedure}

function Find_best_condition
(input $c$ - given attribute; $S$ - set of examples; output best_t - bestcondition)
begin
    best_t := $\emptyset$;
    if $c$ is a numerical attribute then
    begin
        $H$:=list of sorted values for attribute $c$ and objects from $S$;
        { $H(i)$ - $i$th unique value in the list }
        for $i$:=1 to length($H$)-1 do
        if object class assignments for $H(i)$ and $H(i+1)$ are different then
        begin
            $v := (H(i) + H(i+1))/2$;
            create a $new\_t$ as either $(c < v)$ or $(c \geq v)$;
            if Better($new\_t, best\_t, criterion$) then $best\_t := new\_t$ ;
        end
    end
    else { attribute is nominal }
    begin
        for each value $v$ of attribute $c$ do
        if Better($(c = v), best\_t, criterion$) then $best\_t := (c = v)$ ;
    end
end {function}.
```

For the evaluation measure (i.e. a function *Better*) indicating the best condition, one can use either *class entropy* measure or *Laplacian* accuracy. For their definitions see [14] or [23]. It is also possible to consider a lexicographic order of two criteria measuring the rule positive cover and, then, its conditional probability (originally considered by Grzymala in his LEM2 algorithm or its last, quite interesting modification called MLEM). In all experiments, presented further in this paper, we will use the entropy as an evaluation measure. Having the best cut-point we choose a condition $(a < v)$ or $(a \geq v)$ that covers more positive examples from the concept B.

In a case of nominal attributes it is also possible to use another option of *Find best condition* function, where a single attribute value in the elementary condition $(a = v_i)$ is extended to a multi-valued set $(a \in W_a)$, where W_a is a subset of values from the attribute domain. This set is constructed in the similar way as in techniques for inducing binary classification trees. Moreover, the author created MODLEM version with another version of rule stop condition. Let us notice that in the above schema the candidate T is accepted to become a rule if $[T] \subseteq B$, i.e. a rule should cover learning examples belonging to an appropriate approximation of the given class K_j. For some data sets – in particular noisy ones – using this stop condition may produce too specific rules (i.e. containing many elementary conditions and covering too few examples). In such situations the user may accept partially discriminating rules with high enough accuracy – this could be done by applying another stop condition $([T \cap B]/[T] \geq \alpha$. An alternative is to induce all, even too specific rules and to post-process them – which is somehow similar to pruning of decision trees.

Finally we can illustrate the use of MODLEM by a simple example. The data table contains examples of 17 decision concerning classification of some customers into three classes coded as d, p, r. All examples are described by 5 qualitative and numerical attributes.

Table 1. A data table containing examples of customer classification

Age	Job	Period	Income	Purpose	Decision
m	u	0	500	K	r
sr	p	2	1400	S	r
m	p	4	2600	M	d
st	p	16	2300	D	d
sr	p	14	1600	M	p
m	u	0	700	W	r
sr	b	0	600	D	r
m	p	3	1400	D	p
sr	p	11	1600	W	d
st	e	0	1100	D	p
m	u	0	1500	D	p
m	b	0	1000	M	r
sr	p	17	2500	S	p
m	b	0	700	D	r
st	p	21	5000	S	d
m	p	5	3700	M	d
m	b	0	800	K	r

This data table is consistent, so lower and upper approximations are the same. The use of MODLEM results in the following set of certain rules (square brackets contain the number of learning examples covered by the rule):

rule 1. if $(Income < 1050)$ then $(Dec = r)$ [6]
rule 2. if $(Age = sr) \wedge (Period < 2.5)$ then $(Dec = r)$ [2]
rule 3. if $(Period \in [3.5, 12.5))$ then $(Dec = d)$ [2]
rule 4. if $(Age = st) \wedge (Job = p)$ then $(Dec = d)$ [3]
rule 5. if $(Age = m) \wedge (Income \in [1050, 2550)$ then $(Dec = p)$ [2]
rule 6. if $(Job = e)$ then $(Dec = p)$ [1]
rule 7. if $(Age = sr) \wedge (Period \geq 12.5)$ then $(Dec = p)$ [2]

Due to the purpose and page limits of this paper we do not show details of MODLEM working steps while looking for a single rule - the reader is referred to the earlier author's papers devoted to this topic only.

3.2 Classification Strategies

Using rule sets to predict class assignment for an unseen object is based on matching the object description to condition parts of decision rules. This may result in unique matching to rules from the single class. However two other ambiguous cases are possible: matching to more rules indicating different classes or the object description does not match any of the rules. In these cases, it is necessary to apply proper strategies to solve these conflict cases. Review of different strategies is given in [37]

In this paper we employ two classification strategies. The first was introduced by Grzymala in LERS [12]. The decision to which class an object belongs to is made on the basis of the following factors: strength and support. The *Strength* is the total number of learning examples correctly classified by the rule during training. The *support* is defined as the sum of scores of all matching rules from the class. The class K_j for which the support, i.e., the following expression

$$\sum_{matching\ rules\ R\ describing\ K_i} Strength\_factor(R)$$

is the largest is the winner and the object is assigned to K_j.

If complete matching is impossible, all partially matching rules are identified. These are rules with at least one elementary condition matching the corresponding object description. For any partially matching rule R, the factor, called *Matching factor* (R), defined as a ratio of matching conditions to all conditions in the rule, is computed. In partial matching, the concept K_j for which the following expression is the largest

$$\sum_{partially\ matching\ rules\ R} Matching\_factor(R) * Strength\_factor(R)$$

is the winner and the object is classified as being a member of K_j.

The other strategy was introduced in [32]. The main difference is in solving no matching case. It is proposed to consider, so called, *nearest rules* instead of partially matched ones. These are rules nearest to the object description in the sense of chosen distance measure. In [32] a weighted heterogeneous metric DR is used which aggregates a normalized distance measure for numerical attributes and {0;1} differences for nominal attributes. Let r be a nearest matched rule, e denotes a classified object. Then $DR(r, e)$ is defined as:

$$Dr(r, e) = \frac{1}{m} (\sum_{a \in P} d_a^p)^{1/p}$$

where p is a coefficient equal to 1 or 2, m is the number of elementary conditions in P – a condition part of rule r. A distance d_a for numerical attributes is equal to $|a(e) - v_{ai}|/|v_{a-max} - v_{a-min}|$, where v_{ai} is the threshold value occurring in this elementary condition and v_{a-max}, v_{a-min} are maximal and minimal values in the domain of this attribute. For nominal attributes present in the elementary condition, distance d_a is equal to 0 if the description of the classified object e satisfies this condition or 1 otherwise. The coefficient expressing rule similarity (complement of the calculated distance, i.e. $1 - DR(r, e)$) is used instead of matching factor in the above formula and again the strongest decision K_j wins. While computing this formula we can use also heuristic of choosing the first k nearest rules only. More details on this strategy the reader can find in papers [32, 33, 37].

Let us consider a simple example of classifying two objects $e_1 = \{(Age = m), (Job = p), (Period = 6), (Income = 3000), (Purpose = K)\}$ and $e_2 = \{(Age = m), (Job = p), (Period = 2), (Income = 2600), (Purpose = M)\}$. The first object is completely matched by to one rule no. 3. So, this object is be assigned to class d. The other object does not satisfy condition part of any rules. If we use the first strategy for solving no matching case, we can notice that object e_2 is partially matched to rules no. 2, 4 and 5. The support for class r is equal to $0.5 \cdot 2 = 1$. The support for class d is equal to $0.5 \cdot 2 + 0.5 \cdot 2 = 2$. So, the object is assigned to class d.

3.3 Summarizing Experience with Single MODLEM Classifiers

Let us shortly summarize the results of studies, where we evaluated the classification performance of the single rule classifier induced by MODLEM. There are some options of using this algorithm. First of all one can choose as decision concepts either lower or upper approximations. We have carried out several experimental studies on benchmark data sets from ML Irvine repository [3]. Due to the limited size of this paper, we do not give precise tables but conclude that generally none of approximations was better. The differences of classification accuracies were usually not significant or depended on the particular data at hand. This observation is consistent with previous experiments on using certain or possible rules in the framework of LEM2 algorithm [13]. We also noticed that using classification strategies while solving ambiguous matching was necessary for all data sets. Again the difference of applied strategies in case of non-matching (either Grzymala's proposal or nearest rules) were not significant. Moreover, in [14] we performed a comparative study of using MODLEM and LEM2 algorithms on numerical data. LEM2 was used with preprocessing phase with the good discretization algorithm. The results showed that MODLEM can achieved good classification accuracy comparable to best pre-discretization and LEM2 rules.

Here, we could comment that elements of rough sets are mainly used in MODLEM as a kind of preprocessing, i.e. approximations are decision concepts. Then, the main procedure of this algorithm follows rather the general inductive principle which is common aspect with many machine learning algorithms – see e.g. a discussion of rule induction presented in [23]. Moreover, the idea of handling numerical attributes is somehow consistent with solutions also already present in classification tree generation. In this sense, other rule generation algorithms popular in rough sets community, as e.g. based on Boolean reasoning, are more connected with rough sets theory.

It is natural to compare performance of MODLEM induced rules against standard machine learning systems. Such a comparative study was carried out in [37, 41] and showed that generally the results obtained by MODLEM (with nearest rules strategies) were very similar to ones obtained by C4.5 decision tree.

4 Combined Classifiers – General Issues

In the next sections we will study the use of MOLDEM in the framework of the combined classifiers. Previous theoretical research (see, e.g., their summary in [6, 45]) indicated that combining several classifiers is effective only if there is a substantial *level of disagreement* among them, i.e. if they make errors independently with respect to one another. In other words, if they make errors for a given object they should indicate different class assignments. Diversified base classifiers can be generated in many ways, for some review see, e.g. [6, 36, 45]. In general, either *homogeneous* or *heterogeneous classifiers* are constructed.

In the first category, the same learning algorithm is used over different samples of the data set. The best-known examples are either *bagging* and *boosting* techniques which manipulate set of examples by including or weighting particular examples, or methods that manipulate set of attributes, e.g. randomly choosing several attribute subsets. Moreover, multiple classifiers could be trained over different samples or partitions of data sets.

In the second category, different learning *algorithms* are applied to the same data set, and the diversity of results comes from heterogeneous knowledge representations or different evaluation criteria used to construct them. The *stacked generalization* or *meta-learning* belong to this category. In section 7 we study the *combiner* as one of these methods.

Combining classification predictions from single classifiers is usually done by *group* or *specialized decision making*. In the first method all base classifiers are consulted to classify a new object while the other method chooses only these classifiers whose are expertised for this object. *Voting* is the most common method used to combine single classifiers. The vote of each classifier may be weighted, e.g., by an evaluation of its classification performance.

Moreover, looking into the rough sets literature one can notice a growing research interest in constructing more complex classification system. First works concerned rather an intelligent integration of different algorithms into *hybrid* system. For instance, some researchers tried to refine rule classifiers by analysing relationships with neural networks [44]. More related works included an integration of k - nearest neighbor with rough sets rule generation, see e.g. RIONA system, which offered good classification performance [8]. Yet another approach comprises two level knowledge representation: rules induced by Explore representing general patterns in data and case base representing exceptions [36], which worked quite well for the difficult task of credit risk prediction [43]. Recently Skowron and his co-operators have been developing *hierarchical classifiers* which attempt at approximating more complex concepts [2]. Classifiers on different hierarchy level correspond to different levels of pattern generalization and seems to be a specific combination of multiple models, which could be obtained in various ways, e.g. using a special lattice theory [46] or leveled rule generation. Nguyen et al. described in [24] an application concerning detecting sunspots where hierarchical classifier is constructed with a domain knowledge containing an ontology of considered concepts.

5 Using MODLEM Inside the Bagging

Firstly, we consider the use of MODLEM induced classifier inside the most popular homogeneous multiple classifiers [38].

This approach was originally introduced by Breiman [4]. It aggregates classifiers generated from different bootstrap samples. The *bootstrap sample* is obtained by uniformly *sampling with replacement* objects from the training set. Each sample has the same size as the original set, however, some examples do not appear in it, while others may appear more than once. For a training set with m examples, the probability of an example being selected at least once is $1-(1-1/m)^m$. For a large m, this is about 1 - $1/e$. Given the parameter R which is the number of repetitions, R bootstrap samples $S_1, S_2, \ldots, S_R$ are generated. From each sample S_i a classifier C_i is induced by the same learning algorithm and the final classifier C^* is formed by aggregating these R classifiers. A final classification of object x is built by a uniform voting scheme on $C_1, C_2, \ldots, C_R$, i.e. is assigned to the class predicted most often by these sub-classifiers, with ties broken arbitrarily. For more details and theoretical justification see e.g. [4].

Table 2. Comparison of classification accuracies [%] obtained by the single MODLEM based classifier and the bagging approach; R denotes the number of component classifiers inside bagging

Name of data set	Single classifier	Bagging	R
bank	93.81 ± 0.94	95.22 ± 1.02	7
buses	97.20 ± 0.94	99.54 ± 1.09	5
zoo	94.64 ± 0.67	93.89* ± 0.71	7
hepatitis	78.62 ± 0.93	84.05 ± 1.1	5
hsv	54.52 ± 1.05	64.78 ± 0.57	7
iris	94.93 ± 0.5	95.06* ± 0.53	5
automobile	85.23 ± 1.1	83.00 ±0.99	5
segmentation	85.71 ± 0.71	87.62 ± 0.55	7
glass	72.41 ± 1.23	76.09 ± 0.68	10
bricks	90.32* ± 0.82	91.21* ± 0.48	7
vote	92.67 ± 0.38	96.01 ± 0.29	10
bupa	65.77 ± 0.6	76.28 ± 0.44	5
election	88.96± 0.54	91.66 ± 0.34	7
urology	63.80 ± 0.73	67.40 ± 0.46	7
german	72.16 ± 0.27	76.2 ± 0.34	5
crx	84.64 ± 0.35	89.42 ± 0.44	10
pima	73.57 ± 0.67	77.87 ± 0.39	7

In this paper we shortly summarize main results obtained in the extensive computational study [38]. The MODLEM algorithm was applied to generate base classifiers in the bagging combined classifier. In table 2 we present the comparison of the classification accuracy obtained for the best variant of the bagging against the single rule classifier (also induced by MODLEM). The experiments were carried out on several data sets coming mainly from ML Irvine repository [3]. For each data set, we show the classification accuracy obtained by a single classifier over the 10 cross-validation loops. A standard deviation is also given. An asterisk indicates that the difference for these compared classifiers and a given data set is not statistically significant (according to two-paired t-Student test with α=0.05). The last column presents the number of R component classifiers inside the bagging - more details on tuning this value are described in [38].

We conclude that results of this experiment showed that the bagging significantly outperformed the single classifier on 14 data sets out of total 18 ones. The difference between classifiers were non-significant on 3 data sets (those which were rather easy to learn as, e.g. *iris* and *bricks* - which were characterized by a linear separation between classes). Moreover, we noticed the slightly worse performance of the bagging for quite small data (e.g. *buses*, *zoo* - which seemed to be too small for sampling), and significantly better for data sets containing a higher number of examples. For some of these data sets we observed an substantial increase of predictive accuracy, e.g. for *hsv* – over 10%, *bupa* – around 10% and *hepatitis* – 5.43%.

However, we should admit that this good performance was expected as we know that there are many previous reports on successful use of decision trees in bagging or boosting.

6 On Solving Multiclass Problems with the n^2-classifier

One can say the bagging experiment has been just a variant of a standard approach. Now we will move to more original approach, called the n^2-classifier, which was introduced by Jelonek and author in [18, 36]. This kind of a multiple classifier is a specialized approach to solve *multiple class learning problems.*

The n^2-classifier is composed of $(n^2 - n)/2$ base binary classifiers (where n is a number of decision classes; $n > 2$). The main idea is to discriminate each pair of the classes: (i, j), $i, j \in [1..n], i \neq j$, by an independent binary classifier C_{ij}. Each base binary classifier C_{ij} corresponds to a pair of two classes i and j only. Therefore, the specificity of the training of each base classifier C_{ij} consists in presenting to it a subset of the entire learning set that contains only examples coming from classes i and j. The classifier C_{ij} yields a binary classification indicating whether a new example $\mathbf{x}$ belongs to class i or to class j. Let us denote by $C_{ij}(\mathbf{x})$ the classification of an example $\mathbf{x}$ by the base classifier C_{ij}.

The complementary classifiers: C_{ij} and C_{ji} (where $i, j \in < 1 \ldots n >$; $i \neq j$) solve the same classification problem − a discrimination between class i-th and j-th. So, they are equivalent ($C_{ij} \equiv C_{ji}$) and it is sufficient to use only (n^2 - n)/2 classifiers $C_{ij}(i < j)$, which correspond to all combinations of pairs of n classes.

An algorithm providing the final classification assumes that a new example $\mathbf{x}$ is applied to all base classifiers C_{ij}. As a result, their binary predictions $C_{ij}(\mathbf{x})$ are computed. The final classification is obtained by an aggregation rule, which is based on finding a class that wins the most pairwise comparisons. The more sophisticated approach includes a *weighted* majority voting rules, where the vote of each classifier is modified by its credibility, which is calculated as its classification performance during learning phase; more details in [18].

We have to remark that the similar approach was independently studied by Friedman [7] and by Hastie and Tibshirani [17] – they called it *classification by pairwise coupling.* The experimental studies, e.g. [7, 17, 18], have shown that such multiple classifiers performed usually better than the standard classifiers. Previously the author and J.Jelonek have also examined the influence of a learning algorithm on the classification performance of the n^2-classifier.

Table 3. Comparison of classification accuracies [%] and computation times [s] for the single MODLEM based classifier and the n^2-classifier also based on decision rules induced by MODLEM algorithm

Name of data set	Accuracy of single MODLEM (%)	Accuracy of n^2_{MODLEM} (%)	Time of comput. MODLEM	Time of comput. n^2_{MODLEM}
automobile	85.25 ± 1.3	87.96 ± 1.5	15.88 ± 0.4	5.22 ± 0.3
cooc	55.57 ± 2.0	59.30 ± 1.4	4148,7 ± 48.8	431.51 ± 1.6
ecoli	79.63 ± 0.8	81.34 ± 1.7	27.53 ± 0.5	11.25 ± 0.7
glass	72.07 ± 1.2	74.82 ± 1.4	45.29 ± 1.1	13.88 ± 0.4
hist	69.36 ± 1.1	73.10 ± 1.4	3563.79 ± 116.1	333.96 ± 0.8
meta-data	47.2 ± 1.3	49.83 ± 1.9	252.59 ± 78.9	276.71 ± 5.21
iris	94.2 ± 0.6	95.53* ± 1.2	0.71 ± 0.04	0.39 ± 0.04
soybean-large	91.09 ± 0.9	91.99* ± 0.8	26.38 ± 0.3	107.5 ± 5.7
vowel	81.81 ± 0.5	83.79 ± 1.2	3750.57 ± 30.4	250.63 ± 0.7
yeast	54.12 ± 0.7	55.74 ± 0.9	1544.3 ± 13.2	673.82 ± 9.4
zoo	94.64 ± 0.5	94.46* ± 0.8	0.30 ± 0.02	0.34 ± 0.12

Here, we summarize these of our previous results, where the MODLEM was applied to generate base classifiers inside the n^2-classifier [38]. In table 3 we present classification accuracies obtained by the n^2-classifier and compare them against the single rule classifier induced by MODLEM on 11 data sets, all concerning multiple-class learning problems, with a number of classes varied from 3 up to 14. The second and third columns are presented in a similar way as in Table 2. These results showed that the n^2-classifier significantly (again in the sense of paired t test with a significance level $\alpha = 0.05$) outperformed the single classifier on 7 out of 11 problems, e.g. for *hist* – over 3.7%, *glass* – around 2.7%, *automobile* – 2.5% and *meta-data* – 2.6%. These improvements were not so high as in the bagging but still they occurred for many difficult multi-class problems. Again, the multiple classifier was not useful for easier problems (e.g. *iris*). Moreover, we noticed that its performance was better for data sets with a higher number of examples. Coming back to our previous results for the n^2-classifier [18] we can again remark that the comparable classification improvements were observed for the case of using decision trees.

Then, let us focus our attention on interesting phenomena concerning computation costs of using the MODLEM in a construction of the n^2-classifier. Table 3 (two last columns) contains computation times (in seconds calculated as average values over 10 folds with standard deviations). We can notice that generally constructing a combined classifiers does not increase the computation time. What is even more astonishing, for some data sets constructing the n^2-classifier requires even less time than training the standard single classifier. Here, we have to stress that in our previous works [18, 37] we noticed that the increase of classification accuracy (for other learning algorithms as e.g. decision trees, k-nearest neighbor or neural networks) was burden with increasing the computational costs (sometimes quite high). In [38] we attempted to explain the good performance of MODLEM inside the n^2-classifier. Shortly speaking, the n^2-classifier should be rather applied to solving difficult ("complex") classification tasks, where examples of decision classes are separated by non-linear decision borders – these are often difficult concepts to be learned by standard classifiers, while pairwise decision boundaries between each pair of classes may be simpler and easier to be learned with using a *smaller number* of attributes. Here, MODLEM could gain its performance thanks to his sequential covering and greedy heuristic search. It generates rules distinguishing smaller number of learning examples (from two classes only) than in the multiple class case and, above all, testing a smaller number of elementary conditions. To verify hypothesis we inspect syntax of rule sets induced by the single classifier and the n^2-classifier. Rules for binary classifiers were using less attributes and covered more learning example on average than rules from the single set generated in the standard way [38].

7 Combining Predictions of Heterogeneous Classifiers

In two previous sections we described the use of MODLEM based classifiers inside the architecture of homogeneous classifiers. In these solutions, the MODLEM was the only algorithm applied to create base classifiers inside multiple classifiers and could directly influence their final performance. Diversification of base classifiers is one of the conditions for improving classification performance of the final system. Let us repeat that in previously considered solutions it was achieved by changing the distribution of examples in the input data.

Another method to obtain component classifier diversity is constructing, so called, *heterogeneous* classifiers. They are generated from the same input data by different learning algorithms which use different representation language and search strategies. These base classifiers could be put inside a layered architecture. At the first level base classifiers receive the original data as input. Their predictions are then aggregated at the second level into the final prediction of the system. This could be done in various ways. In one of our studies we used a solution coming from Chan & Stolfo [5], called a *combiner*.

The combiner is based on an idea of merging predictions of base classifiers by an additional classifier, called *meta-classifier*. This is constructed in an *extra meta-learning* step, i.e. first base classifiers are learned, then their predictions made on a set of extra validation examples, together with correct decision labels, form a meta-level training set. An extra learning algorithm is applied to this set to discover how to merge base classifier predictions into a final decision.

Table 4. Classification accuracies [%] for different multiple classifiers

Data set	Bagging	n^2-classifier	Combiner
Automobile	83.00	87.90	84.90
Bank	95.22	–	95.45
Bupa	76.28	–	69.12
Ecoli	85.70	81.34	85.42
Glass	74.82	74.82	71.50
HSV	64.75	–	59.02
Meta-data	48.11	49.80	51.33
Pima	75.78	–	74.78
Voting	93.33	–	94.67
Yeast	58.18	55.74	58.36
Zoo	93.89	94.46	95.05

In [41] we performed a comparative study of using a combiner approach against the single classifiers learned by these algorithms which were applied to create its component classifiers. In this study base classifiers were induced by k-NN, C4.5 and MODLEM. The meta-classifier was a Naive Bayes. This comparative study was performed on 15 data sets. However, the obtained results showed that the combiner did not improve classification accuracy in so many cases as previously studied homogeneous classifiers. Only in 33% data we observed a significant improvement comparing against single classifiers. In table 4 we present only some of these results concerning the final evaluation of the combiner compared also against the previous multiple classifiers. However, while comparing these classifiers we should be cautious as the number of the results on common data sets was limited. Moreover, MODLEM is only one of three component classifiers inside the combiner that influences the final result.

We could also ask a question about other elements of the architecture of heterogeneous classifier, e.g. number of component classifiers or the aggregation techniques. In recent experiments we focus our interest on testing two other techniques instead of the meta-combiner:

– a simple aggregation performed by means of a majority voting rule (denoted as MV in table 4),
– using a quite sophisticated approach – SCANN; It was introduced by Merz [22] and uses a mechanism of the correspondence analysis to discover hidden relationships between the learning examples and the classification done by the component classifiers.

Results from ongoing experiments are given in Table 5. There is also a difference to previous architecture, i.e. adding an additional, forth component clas-

Table 5. Comparison of different methods producing the final decision inside the heterogeneous classifiers - classification accuracies [%]

Data set	MV	SCANN	Combiner
credit-a	86.2 ± 0.6	87 ± 0.7	86.6 ± 0.4
glass	68.5 ± 0.3	70.1 ± 0.2	70.5 ± 0.6
ecoli	86.1 ± 0.9	81.5 ± 0.8	84.5 ± 0.5
zoo	95 ± 0.9	92.2 ± 0.7	95.1 ± 0.4

sifiers Naive Bayesian at the first level. We can remark that the more advanced aggregation technique could slightly increase the classification accuracy comparing to simpler one. On the other hand they are much time consuming.

8 Discussion of results and final remarks

As Professor Zdzisław Pawlak wrote in the introductory chapter of his book on rough sets [26] knowledge of human beings and other species is strictly connected with their ability to classify objects. Finding classification patterns of sensor signals or data form fundamental mechanisms for very living being. In his point of view it was then connected with a partition (classification) operation leading to basic blocks for constructing knowledge. Many researchers followed the Pawlak's idea. One of the main research directions includes constructing approximations of knowledge from tables containing examples of decisions on object classification. Rules were often induced as the most popular knowledge representation. They could be used either to describe the characteristics of available data or as the basis for supporting classification decisions concerning new objects. Up to now several efficient rule classifiers have been introduced.

In this study we have attempted to briefly describe the current experience with using the author's rule induction algorithm MODLEM, which induces either certain or possible rules from appropriate rough approximations. This is the main point where elements of the rough sets theory is applied in this algorithm. Given as an input learning examples from approximations, the rule generation phase follows the general idea of sequential covering, which is somehow in common with machine learning paradigms. The MODLEM produces a minimal set of rules covering examples from rough approximations. This rule sets should be joined with classification strategies for solving ambiguous matching of the new object description to condition parts of rules. An extra property of this algorithm is it ability to handle directly numerical attributes without prior discretization. The current experience with comparative studies on benchmark data sets and real life applications showed that the classification performance of this approach was comparable to other symbolic classifiers, in particular to decision trees.

Although the MODLEM classifier and other machine learning approaches are efficient for many classification problems, they do not always lead to satisfactory classification accuracy for more complex and difficult problems. This is our motivation to consider new approaches for increasing classification accuracy by combining several classifiers into an integrated system. Several proposals of constructing such multiple classifiers are already proposed. Most of them are general approaches, where many different algorithms could be applied to induce the component classifiers.

Thus, our main research interest in this study is to summarize our experiments with using MODLEM induced rule classifiers inside the framework of three different multiple classifiers, namely the bagging, the n^2-classifier and the combiner. A classification accuracy for the multiple classifier has been compared against the standard classifiers – also induced by MODLEM. These results and their detailed discussion has been given in the previous sections.

Firstly we could notice that using MODLEM inside the bagging was quite effective. However, it was a kind of standard approach and we could expect such good performance as MODLEM performs similarly to decision trees (which have been extensively studied in the bagging) and could be seen as *unstable* learning algorithm - i.e. an algorithm whose output classifier undergoes changes in response to small changes in the training data. This kind of algorithm may produce base classifiers diversified enough (but not too much, see e.g. discussion of experimental study by Kuncheva and Whitaker [21]) which is a necessary condition for their effective aggregation. Following the same arguments we also suspect that MODLEM should nicely work inside the boosting classifier. Further on, we could hypothesize that slightly worse improvements of the classification accuracy in the combiner approach may result from insufficient diversification of component heterogeneous classifiers. This has been verified by analysing distributions of wrong decisions for base classifiers, presented in [41]. It showed the correlation of errors for some data sets, where finally we did not notice the improvement of the classification accuracy.

The most original methodological approach is Jelonek and author's proposal of the n^2-classifier which is in fact a specialized approach to learning multiple class problems. The n^2-classifier is particularly well suited for multiple class data where exist "simpler" pairwise decision boundaries between pair of classes. MODLEM seems to be a good choice to be used inside this framework as it leads to an improvement of classification performance and does not increase computational costs - reasons for this have been discussed in section 7. Let us notice that using other learning algorithms inside the n^2-classifier and applying MODLEM in two other multiple classifier requires an extra computation efforts comparing to learning the single, standard classifier [38].

Comparing results of all together multiple classifiers "head to head" we should be cautious as we had a limited number of common data sets. It seems that the n^2-classifier is slightly better for these data. While the standard multiple classifiers, as bagging or combiner, are quite efficient for simpler data and are easier to be implemented.

To sum up, the results of our experiments have shown that the MODLEM algorithm can be efficiently used within the framework of three multiple classifiers for data sets concerning more "complex" decision concepts. However, the relative merits of these new approaches depends on the specifies of particular problems and a training sample size.

Let us notice that there is a disadvantage of the multiple classifiers - loosing a simple and easy interpretable structure of knowledge represented in a form decision rules. These are ensembles of diversified rule sets specialized for predictive aims not one set of rules in a form for a human inspection.

As to future research directions we could consider yet another way of obtaining diversified data – i.e. selecting different subsets of attributes for each component classifiers. The author has already started research on extending bootstrap samples inside the bagging by applying additionally attribute selection [39, 40]. In this way each bootstrap is replicated few times, each of them using different subset of attributes. We have considered the use of different selection techniques and observed that besides random choice or wrapper model, techniques which use either entropy based measures or correlation merits are quite useful. The results of comparative experiments carried out in [40] have showed that the classification accuracy of such a new extended bagging is higher than for standard one. In this context one could come back to the classical rough sets topic of reducts, which relates to finding an ensemble of few attribute subsets covering different data properties and constructing in this way a set of diversified examples for an integrated system. However, we are not limited to "classical" meaning of pure rough sets reducts but rather to approximate ones, where the entropy measure is also considered [30].

Acknowledgment: The author would like to thank his colleagues Jacek Jelonek, Sławomir Nowaczyk and his M.Sc. students Michał Brończyk, Ryszard Gizelski, Maciej Łuszczyński who have worked with him on the software implementations of the classifiers or took part in some experiments.

References

1. Bazan J.: A comparison of dynamic and non-dynamic rough set methods for extracting laws from decision tables. In Polkowski L., Skowron A. (eds.), *Rough Sets in Data Mining and Knowledge Discovery* vol. 1, Physica-Verlag, 1998, 321–365.
2. Bazan J., Nguyen H.Son, Skowron A.: Rough sets methods in approximation of hierarchical concepts. In *Proc. of the Conference on Rough Sets and New Trends in Computing, RSCTC* – 2004, LNAI 2066, Springer Verlag, 2004, 346–355.
3. Blake C., Koegh E., Mertz C.J.: Repository of Machine Learning, University of California at Irvine (1999).
4. Breiman L.: Bagging predictors. *Machine Learning*, 24 (2), (1996) 123–140.
5. Chan P.K., Stolfo S.: On the accuracy of meta-learning for scalable data mining. *Journal of Intelligent Information Systems*, **8**, (1), 1997, 5-28.
6. Dietrich T.G.: Ensemble methods in machine learning. In *Proc. of 1st Int. Workshop on Multiple Classifier Systems*, 2000, 1–15.
7. Friedman J.: Another approach to polychotomous classification, Technical Report, Stanford University, 1996.
8. Góra G., Wojna A.: RIONA: a new classification system combining rule induction and instance based learning. *Fundamenta Informaticae* **51** (4), 2002, 369-390.

9. Greco S., Matarazzo B., Słowiński R.: The use of rough sets and fuzzy sets in MCDM. In Gal T., Stewart T., Hanne T. (eds), *Advances in Multiple Criteria Decision Making*, Kluwer, chapter 14, 1999, pp. 14.1-14.59.
10. Greco S., Matarazzo B., Słowiński R., Stefanowski J.: Variable consistency model of dominance-based rough set approach. In *Proc. 2nd Int. Conference on Rough Sets and New Trends in Computing, RSCTC* – 2000, LNAI 2005, Springer Verag, 2001,170–181.
11. Grzymala-Busse J.W. LERS - a system for learning from examples based on rough sets. In Slowinski R. (ed.), *Intelligent Decision Support*, Kluwer Academic Publishers, 1992, 3–18.
12. Grzymala-Busse J.W.: Managing uncertainty in machine learning from examples. In *Proc. 3rd Int. Symp. in Intelligent Systems*, Wigry, Poland, IPI PAN Press, 1994, 70–84.
13. Grzymala-Busse J.W. Zou X.: Classification strategies using certain and possible rules. In *Proceedings of the 1th Rough Sets and Current Trends in Computing Conference, RSCTC*–98 , LNAI 1424, Springer Verlag, 1998, 37-44.
14. Grzymala-Busse J.W., Stefanowski J.: Three approaches to numerical attribute discretization for rule induction. *International Journal of Intelligent Systems*, 16 (1), (2001) 29–38.
15. Grzymala-Busse J.W., Stefanowski J. Wilk Sz.: *A comparison of two approaches to data mining from imbalanced data.* In Proc. of the KES 2004 - 8-th Int. Conf. on Knowledge-based Intelligent Information & Engineering Systems, Springer LNCS vol. **3213**, 2004, 757-763.
16. Han J., Kamber M.: *Data mining: Concepts and techniques*, San Francisco, Morgan Kaufmann, 2000.
17. Hastie T., Tibshirani R.: Classification by pairwise coupling. In Jordan M.I. (ed.) *Advances in Neural Information Processing Systems*: 10 (NIPS-97), MIT Press, 1998, 507-513.
18. Jelonek J., Stefanowski J.: Experiments on solving multiclass learning problems by the n^2-classifier. In *Proceedings of 10th European Conference on Machine Learning* ECML 98, Springer LNAI no. 1398, 1998, 172–177.
19. Klosgen W., Żytkow J.M. (eds.): *Handbook of Data Mining and Knowledge Discovery*, Oxford Press, 2002.
20. Komorowski J., Pawlak Z., Polkowski L. Skowron A.: Rough Sets: tutorial. In Pal S.K., Skowron A. (eds) *Rough Fuzzy Hybridization. A new trend in decision making*, Springer Verlag, Singapore, 1999, 3–98.
21. Kuncheva L., Whitaker C.J.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51, 2003, 181–207.
22. Merz C.: Using correspondence analysis to combine classifiers. *Machine Learning*, 36 (1/2), 1999, 33–58.
23. Mitchell Tom M.: *Machine learning*, McGraw Hill, 1997.
24. Nguyen Sinh Hoa, Trung Tham Nguyen, Nguyen Hung Son: Rough sets approach to sunspot classification problem. In *Proc. of the Conference RSFDGrC* – 2005, vol 2, LNAI 3642, Springer Verlag, 2005, 263-272.
25. Pawlak Z.: Rough sets. *Int. J. Computer and Information Sci.*, 11, 1982, 341–356.
26. Pawlak Z.: *Rough sets. Theoretical aspects of reasoning about data.* Kluwer Academic Publishers, Dordrecht, 1991.
27. Pawlak Z., Grzymala-Busse J., Slowinski R., Ziarko W.: Rough sets. *Communications of the ACM*, vol. 38, no. 11, 1995, 89-95.

28. Skowron A.: Boolean reasoning for decision rules generation. In Komorowski J., Ras Z. (des.) *Methodologies for Intelligent Systems*, LNAI 689, Springer-Verlag, 1993, 295–305.
29. Skowron A., Rauszer C.: The discernibility matrices and functions in information systems. In Slowinski R. (ed.), *Intelligent Decision Support. Handbook of Applications and Advances of Rough Set Theory.* Kluwer Academic Publishers, 1992, 331–362 .
30. Slezak D.: Approximate entropy reducts. *Fundamenta Informaticae* **53** (3/4), 2002, 365-387.
31. Slowinski R., Greco S.: Inducing Robust Decision Rules from Rough Approximations of a Preference Relation. In Rutkowski L. et al. (eds): Artiffcial Intelligence and Soft Computing, LNAI 3070, Springer-Verlag, 2004, 118-132.
32. Stefanowski J.: Classification support based on the rough sets. *Foundations of Computing and Decision Sciences*, vol. 18, no. 3-4, 1993, 371-380.
33. Stefanowski J.: Using valued closeness relation in classification support of new objects. In Lin T. Y., Wildberger (eds) *Soft computing: rough sets, fuzzy logic, neural networks uncertainty management, knowledge discovery*, Simulation Councils Inc., San Diego CA, 1995, 324–327.
34. Stefanowski J.: On rough set based approaches to induction of decision rules. In Polkowski L., Skowron A. (eds), *Rough Sets in Data Mining and Knowledge Discovery*, vol. 1, Physica-Verlag, 1998, 500–529.
35. Stefanowski J.: The rough set based rule induction technique for classification problems. In *Proceedings of 6th European Conference on Intelligent Techniques and Soft Computing* EUFIT 98, Aachen 7-10 Sept., 1998, 109–113.
36. Stefanowski J.: Multiple and hybrid classifiers. In Polkowski L. (ed.) *Formal Methods and Intelligent Techniques in Control, Decision Making, Multimedia and Robotics*, Post-Proceedings of 2nd Int. Conference, Warszawa, 2001, 174–188.
37. Stefanowski J.: Algorithims of rule induction for knowledge discovery. (In Polish), Habilitation Thesis published as Series Rozprawy no. 361, Poznan Univeristy of Technology Press, Poznan (2001).
38. Stefanowski J.: The bagging and n2-classifiers based on rules induced by MODLEM. In *Proceedings of the 4th Int. Conference Rough Sets and Current Trends in Computing*, RSCTC – 2004, LNAI 3066, Springer-Verlag, 2004, 488-497.
39. Stefanowski J.: An experimental study of methods combining multiple classifiers - diversified both by feature selection and bootstrap sampling. In K.T. Atanassov, J. Kacprzyk, M. Krawczak, E. Szmidt (eds), *Issues in the Representation and Processing of Uncertain and Imprecise Information*, Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2005, 337-354.
40. Stefanowski J., Kaczmarek M.: Integrating attribute selection to improve accuracy of bagging classifiers. In *Proc. of the AI-METH 2004. Recent Developments in Artificial Intelligence Methods*, Gliwice, 2004, 263-268.
41. Stefanowski J., Nowaczyk S.: On using rule induction in multiple classifiers with a combiner aggregation strategy. In *Proc. of the 5th Int. Conference on Intelligent Systems Design and Applications* - ISDA 2005, IEEE Press, 432-437.
42. Stefanowski J., Vanderpooten D.: Induction of decision rules in classification and discovery-oriented perspectives. *International Journal of Intelligent Systems* **16** (1), 2001, 13–28.
43. Stefanowski J., Wilk S.: Evaluating business credit risk by means of approach integrating decision rules and case based learning. *International Journal of Intelligent Systems in Accounting, Finance and Management* **10** (2001) 97–114.

44. Szczuka M: Refining classifiers with neural networks. *International Journal of Intelligent Systems* **16** (1), 2001, 39–56.
45. Valentini G., Masuli F.: Ensambles of learning machines. In R. Tagliaferri, M. Marinaro (eds), *Neural Nets WIRN Vietri-2002*, Springer-Verlag LNCS, vol. 2486, 2002 , 3–19.
46. Wang H., Duntsch I., Gediga G., Skowron A.: Hyperrelations in version space. *International Journal of Approximate Reasoning*, **23**, 2000, 111–136.
47. Ziarko W.: Variable precision rough sets model. *Journal of Computer and Systems Sciences*, vol. 46. no. 1, 1993, 39–59.