RIONA: A Classifier Combining Rule Induction and k-NN Method with Automated Selection of Optimal Neighbourhood

Grzegorz Góra and Arkadiusz Wojna

Institute of Informatics, Warsaw University
ul. Banacha 2, 02-097 Warsaw, Poland
{ggora, wojna}@mimuw.edu.pl

Abstract. The article describes a method combining two widely-used empirical approaches: rule induction and instance-based learning. In our algorithm (*RIONA*) decision is predicted not on the basis of the whole support set of all rules matching a test case, but the support set restricted to a neighbourhood of a test case. The size of the optimal neighbourhood is automatically induced during the learning phase. The empirical study shows the interesting fact that it is enough to consider a small neighbourhood to preserve classification accuracy. The combination of *k-NN* and a rule-based algorithm results in a significant acceleration of the algorithm using all minimal rules. We study the significance of different components of the presented method and compare its accuracy to well-known methods.

1 Introduction

Many techniques of inductive concept learning from its instances have been developed so far [10]. Empirical comparison of these approaches shows that each performs well on some, but not all, domains. A great progress has been made in multistrategy learning to combine these approaches in order to construct a classifier that has properties of two or more techniques. Although the problem of inductive generalisation has no general solution (what is known as the conservation law for generalisation performance [11]), the goal is to increase the average accuracy for the real-world domains at the expense of accuracy for the domains that never occur in practice.

We present a multi-strategy learning approach combining the rule induction [9] and the instance-based techniques [3], [5]. There has been a lot of work done in this area [4], [6], [7]. Our algorithm considers all minimal decision rules, i.e. the most general rules consistent with training examples. It simulates classification based on the most frequent class in the support set of minimal rules covering a test object. The main idea is that the support set is restricted to the neighbourhood of a test example. The neighbourhood of a test example consists of either the objects within some distance from a test example or a number of objects closest to a test example (like in *k-NN* method). The appropriate size of

a neighbourhood to be taken for classification is automatically induced during the process of learning. The crucial empirical observation is that taking a neighbourhood that is much smaller than the whole training set preserves or even improves accuracy. It enables both to induce the optimal neighbourhood during the learning phase and to classify objects effectively.

The paper is organised as follows. In Section 2 the paper will be placed in the context of related work. Section 3 outlines the main features of two techniques that are most relevant to this work, i.e. rule induction and instance-based learning. Our algorithm, combining these approaches, is presented in Section 4. Section 5 provides experimental results evaluating the accuracy and the speed of the presented system. Section 6 concludes this paper with a brief summary and discussion of possible directions for future research.

2 Related Work

In recent literature there has been a number of works combining instance-based and decision rule induction methods.

RISE system [4] is based on unification of these two methods. The difference between *RISE* system and our approach is that *RISE* selects the class for a test object on the basis of the closest rule. First, *RISE* generates decision rules. At the beginning instances are treated as maximally specific rules and these rules are then gradually generalised as long as the global leave-one-out accuracy is improving. An object is classified according to the closest rule. The distance between an object and a rule is measured with the metric combining the normalised Manhattan metric for numerical attributes and the Simple Value Difference Metric (SVDM) for symbolic attributes.

An approach more similar to our method is presented in *DeEPs* and *DeEPsNN* [7]. The first difference is that *DeEPs* uses a different form of rule conditions and different criteria for rule selection. *DeEPs* classifies objects on the basis of all rules that have high frequency-changing rate (a measure similar to confidence). While classifying a test object the system computes the support set using all rules with high frequency-changing rate and selects the most frequent class in the support set. In our system the computed support set is limited to a certain neighbourhood of a test object. *DeEPsNN* combines *3-NN* and *DeEPs*: if a certain fixed neighbourhood of a test object covers at least one training object, *3-NN* is applied, otherwise *DeEPs* is used.

In [1] an algorithm with the lazy rule induction approach is presented. It computes the support set of all minimal rules covering a test object in the following way. For each training object the algorithm constructs the local rule containing the common conditions of the test and the training objects and checks whether the training objects supporting the local rule are in the same decision class. Finally, the algorithm selects the class most frequent in the support set. This algorithm treats all attributes as symbolic. We generalised this approach for symbolic attributes and extended it to numerical attributes.

A detailed study of *k-NN* algorithms is presented in [12]. In particular, that paper describes research on selection of the optimal value of k. The experiments presented in that paper showed that the accuracy of *k-NN* is insensitive to the exact choice of k when the optimal k is large enough. Different methods for adapting the value of k locally within different parts of the input space have also been investigated. The local selection of k improves accuracy for data that contain noise or irrelevant features.

Our approach combines the idea used in [1] (extended as described above) with *k-NN* method in such a way that it considers local rules only for the training examples from the k-nearest neighbourhood of a test example. The distance is measured with the metric used in *RISE* [4]. Moreover, the algorithm searches for the global optimal value k during the learning phase. This combination improves the accuracy of a *k-NN* classifier with a fixed value k and helps to reach the accuracy comparable to a rule-based classifier in case when the accuracy of the *k-NN* method is low.

3 Preliminaries and Definitions

We assume that a training set, denoted in the paper *trnSet*, consists of a finite set of examples. Each example is described by a finite set of attributes (features) $A \cup \{d\}$, i.e. $a : trnSet \to V_a$ for $a \in A \cup \{d\}$, where $d \notin A$ denotes the decision attribute and V_a is a value domain of the attribute a. Two groups of attributes are considered: symbolic and numerical (real-valued). We denote by $Class(v)$ a subset of training examples with a class v. We also assume that $V_d = \{1, ..., |V_d|\}$.

3.1 Minimal and Lazy Rule Induction

Rule induction algorithms induce decision rules from a training set. A decision rule consists of a conjunction of attribute conditions and a consequent. The commonly used conditions are equations $attribute = value$ for symbolic attributes and interval inclusion for numerical attributes, e.g. $IF\ (a_1 = 2 \wedge a_3 \in [3, 7] \wedge a_6 = 5)\ THEN\ (d = 1)$.

Many systems compute a set of such decision rules and then use it in the classification process. Another approach is the lazy concept induction that does not require calculation of decision rules before classification of new objects. An example of such an algorithm is presented in [1]. It generates only decision rules relevant for a new test object and then classifies it like algorithms generating rules in advance. Below we briefly describe this algorithm generalised for symbolic attributes and extended to the case of numerical attributes.

Definition 1. *For objects* tst, trn *we denote by* $\mathrm{rule}_{\mathrm{tst}}(\mathrm{trn})$ *the local rule with decision* $d(\mathrm{trn})$ *and the following conditions* c_i *for each attribute* a_i*:*

$$c_i = \begin{cases} a_i \in [\min(a_i(\mathrm{tst}), a_i(\mathrm{trn})), \max(a_i(\mathrm{tst}), a_i(\mathrm{trn}))] & \text{when } a_i \text{ is numerical} \\ a_i \in B\,(a_i(\mathrm{tst}), \delta(a_i(\mathrm{tst}), a_i(\mathrm{trn}))) & \text{when } a_i \text{ is symbolic} \end{cases}$$

where $B(c, R)$ *is a ball centered in* c *with radius* R *and* δ *is a measure of attribute value similarity.*

The conditions in Definition 1 are chosen so that both the training and the test example satisfy the rule and the conditions are maximally specific. The condition used in [1] is a particular case of the above condition defined for symbolic attributes when Hamming metric is used ($\delta(x,y) = 1$ if $x \neq y$ and 0 otherwise). Below we present the lazy rule induction algorithm (*RIA*). The function *isConsistent(r,verifySet)* checks whether a local rule r is consistent with a *verifySet*.

Algorithm 1 RIA(tst)

1. ***for*** *each class* $v \in V_d$
2. $\quad \text{supp}(v) = \emptyset$
3. $\quad$ ***for*** *each* $\text{trn} \in \text{trnSet}$ *with* $d(\text{trn}) = v$
4. $\quad\quad$ ***if*** $\text{isConsistent}(\text{rule}_{\text{tst}}(\text{trn}), \text{trnSet})$
5. $\quad\quad\quad$ ***then*** $\text{supp}(v) = \text{supp}(v) \cup \{\text{trn}\}$
6. $\text{RIA} = \arg\max_{v \in V_d} \frac{|\text{supp}(v)|}{|\text{Class}(v)|}$

It was shown in [1] that *RIA* is equivalent to the algorithm based on calculating all rules that are maximally general and consistent with the training set. The time complexity of *RIA* for a single test object is $O(n^2)$, where $n = |trnSet|$. One of the motivations behind our work was to reduce this complexity.

3.2 Instance-Based Learning

A commonly used instance-based learning method is the k nearest neighbours algorithm (*k-NN*). It is based on the concept of similarity. Given a number of training examples the class for a test case is inferred from the k nearest examples in the sense of a similarity measure. Different measures are used for numerical and symbolic domains. For domains with both types of attributes a combination of these approaches may be used:

$$\varrho(x,y) = \sum_{a \in A} \delta_a(x,y)$$

where x, y are objects and $\delta_a(\cdot,\cdot)$ is a measure of attribute value similarity. In the paper we used the normalised Manhattan distance for numerical attributes and SVDM (see e.g. [4]) for symbolic attributes:

$$\delta_a(x,y) = \begin{cases} \left| \frac{a(x)-a(y)}{a^{\max}-a^{\min}} \right| & \text{for } a \text{ - numerical} \\ \sum_{v \in V_d} |P(Class(v)|a(x)) - P(Class(v)|a(y))| & \text{for } a \text{ - symbolic} \end{cases}$$

4 Rule Induction with Optimal Neighbourhood Algorithm (RIONA)

Instead of considering all training examples in building a support set like in *RIA*, we can limit it to a certain neighbourhood of a test example. The intuition behind it is that training examples far from a test object are less relevant for classification than closer examples. We consider two classes of a neighbourhood:

Definition 2. *For each test example* tst *we define* $S(\mathrm{tst}, k)$ *as the set of* k *training examples that are most similar to* tst *according to a similarity measure* ϱ.

Definition 3. *For each test example* tst *we define* $B(\mathrm{tst}, R)$ *as the set of training examples* trn *such that* $\varrho(\mathrm{tst}, \mathrm{trn}) \leq R$.

The former neighbourhood is similar to the one used in the *k-NN* algorithm. From now on, we use in the paper $S(tst, k)$ neighbourhood, although we studied both classes of neighbourhoods in parallel and the empirical difference between them will be discussed in Section 5.

Now we are ready to present an approach to induction that is a kind of combination of case-based learning (see Section 3.2) and lazy minimal rule induction (see Section 3.1). The main idea is that we apply the following strategy for conflict resolving:

$$NormNStrength(tst, v) = \frac{\left| \bigcup_{r \in MinRules^{v}_{tst}} supp(r) \cap S(tst, k) \right|}{|Class(v)|} \tag{1}$$

where v denotes the v-th class, *tst* is a test example, $supp(r)$ is the set of training examples matching a rule r, $MinRules^{v}_{tst}$ is the set of all rules maximally general and consistent with the training set, whose premise is satisfied by *tst* and the consequent is the class v.

In the classification process we assume that the parameter k of the neighbourhood is fixed. The proper size of the neighbourhood is found in the learning phase (see Section 4.1).

In order to calculate the measure (1) we used a modified version of Algorithm 1. First, in the line 3 of the algorithm only the examples $trn \in S(tst, k)$ should be considered. Furthermore, it is not necessary to consider all the examples from the training set to check the consistency of the $rule_{tst}(trn)$. Please note that from Definition 1 we have that:

Proposition 1. *If* trn' *satisfies* $\mathrm{rule}_{\mathrm{tst}}(\mathrm{trn})$ *then* $\varrho(\mathrm{tst}, trn') \leq \varrho(\mathrm{tst}, \mathrm{trn})$.

Hence, the examples that are distanced from the test example *tst* more than the training example *trn* can not cause inconsistency of $rule_{tst}(trn)$. The resulting classification algorithm is presented below. It predicts the most common class among the training examples that are covered by the rules satisfied by a test example and that are in the specified neighbourhood.

Algorithm 2 RIONA(tst)
neighbourSet $= S(\mathrm{tst}, k)$
for *each class* $v \in V_d$
supp$(v) = \emptyset$
for *each* trn $\in$ neighbourSet *with* $d(\mathrm{trn}) = v$
if isConsistent$(\mathrm{rule}_{\mathrm{tst}}(\mathrm{trn}), \mathrm{neighbourSet})$
then supp$(v) =$ supp$(v) \cup \{\mathrm{trn}\}$
RIONA $= \arg\max_{v \in V_d} \frac{|\mathrm{supp}(v)|}{|\mathrm{Class}(v)|}$

For the maximal neighbourhood the algorithm *RIONA* works exactly as *RIA* algorithm. On the other hand, taking a neighbourhood as a single nearest training example we obtain the nearest neighbour algorithm. In this sense *RIONA* belongs between the nearest neighbour and the rule induction classifier.

4.1 Selection of Optimal Neighbourhood

During the experiments (see Section 5) we found that the performance of the algorithm can significantly depend on the size of a chosen neighbourhood and a different size is appropriate for different problem domains. In fact, it is possible to estimate the optimal value k for $S(tst, k)$ neighbourhood. It would be similar if the optimal value k for *k-NN* method were estimated. The idea is that one can use the leave-one-out method on a training set to estimate the accuracy of the classifier for different values of k ($1 \leq k \leq k_{max}$) and then choose the value k for which the estimation is the greatest. Applying it directly would require repeating the leave-one-out estimation k_{max} times. However, we emulated this process in a time comparable to the single leave-one-out test for k equal to the maximal possible value $k = k_{max}$. This idea is realised in Algorithm 3.

Algorithm 3 findOptimalK(k_{max})
  for *each* trn $\in$ trnSet $A_{\text{trn}} = \text{getClassificationVector}(\text{trn}, k_{max})$
  return $\arg\max_k |\{\text{trn} \in \text{trnSet} : d(\text{trn}) = A_{\text{trn}}[k]\}|$
  function getClassificationVector(tst, k_{max})
  NN = *vector of* k_{max} *training examples* $NN_1, \ldots, NN_{k_{max}}$
    nearest to tst *sorted according to a distance* $\varrho(\text{tst}, \cdot)$
  for *each class* $v \in V_d$ decStrength[v] = 0
  currentDec= *the most frequent class in* trnSet
  for $k = 1, 2, ..., k_{max}$
    if isConsistent(rule$_{\text{tst}}(NN_k), NN)$ ***then***
      $v = d(NN_k)$
      decStrength[v] = decStrength[v] + 1
      if $\frac{\text{decStrength}[v]}{|\text{Class}(v)|} > \frac{\text{decStrength[currentDec]}}{|\text{Class(currentDec)}|}$ ***then*** currentDec = v
    $D[k]$ = currentDec
  return D

Ignoring the consistency checking in the function $getClassificationVector(\cdot, \cdot)$ we obtain the k nearest neighbours algorithm with selection of the optimal k (*ONN*). An experimental comparison of *RIONA* and *ONN* is presented in the next section.

5 Experimental Study

Table 1 presents experimental results for 24 data sets from UCI repository [2]. For data that are split into a training and a testing set the experiments were performed for joined data. The accuracy for *C5.0*, *DeEPs* and *DeEPsNN* are taken

Table 1. The average optimal k, the average accuracy (%) and the standard deviation for *RIONA* with the optimal k-best neighbourhood and the average accuracy (%) for the other systems: *RIA*, *ONN*, *3-NN*, *RIONA* with the optimal $B(tst, R)$ neighbourhood, *C5.0*, *DeEPs* and *DeEPsNN*. The superscripts denote the confidence levels: 5 is 99.9%, 4 is 99%, 3 is 97.5%, 2 is 95%, 1 is 90%, and 0 is below 90%. Plus indicates that the average accuracy of an algorithm is higher than in *RIONA* and minus otherwise

Domain (size, attr, classes)	k_{opt}	RIONA	RIA	ONN	3-NN	RIONA(B)	C5.0	DeEPs	DeEPsNN
australian (690, 14, 2)	41,2	86,1±0,4	$65,0^{-5}$	$85,7^{-2}$	$85,0^{-4}$	$85,7^{-2}$	85,9	84,9	**88,4**
breast (277, 9, 2)	77,9	73,4±1,0	**73,9**^{0}	$73,0^{0}$	$68,6^{-5}$	$73,6^{0}$	-	-	-
breast-wis (683, 9, 2)	3,0	97,0±0,3	$89,7^{-5}$	$97,0^{0}$	**97,1**^{0}	$96,1^{-5}$	95,4	96,4	96,3
bupa-liver (345, 6, 2)	40,6	**66,6±1,7**	$63,0^{-5}$	$64,1^{-4}$	$66,0^{0}$	$66,4^{0}$	-	-	-
census (45222, 16, 2)	42,1	83,8±0,0	-	$84,1^{+5}$	$82,0^{-5}$	$83,9^{+5}$	85,8	**85,9**	**85,9**
chess (3196, 36, 2)	11,9	98,0±0,1	-	$96,9^{-5}$	$97,0^{-5}$	$97,5^{-5}$	**99,4**	97,8	97,8
german (1000, 20, 2)	29,2	**74,5±0,5**	$70,1^{-5}$	$74,1^{-1}$	$72,1^{-5}$	$73,1^{-4}$	71,3	74,4	74,4
glass (214, 9, 6)	2,1	70,7±1,9	$39,5^{-5}$	$70,7^{0}$	**71,9**$^{+1}$	$63,9^{-5}$	70,0	58,5	68,0
heart (270, 13, 2)	19,4	83,2±1,0	$62,8^{-5}$	$83,1^{0}$	$81,3^{-5}$	**83,4**^{0}	77,1	81,1	81,1
iris (150, 4, 3)	37,1	94,6±0,6	$90,5^{-5}$	$94,4^{0}$	$95,3^{+4}$	$94,7^{0}$	94,0	**96,0**	**96,0**
letter (20000, 16, 26)	3,8	**95,8±0,1**	-	**95,8**^{0}	**95,8**^{0}	$94,0^{-5}$	88,1	93,6	95,5
lymph (148, 18, 4)	1,4	85,4±1,3	$76,4^{-5}$	**86,3**$^{+1}$	$84,4^{-2}$	$81,4^{-5}$	74,9	75,4	84,1
mushroom (8124, 22, 2)	1,0	**100,0±0,0**	-	**100,0**^{0}	**100,0**^{0}	**100,0**^{0}	**100,0**	**100,0**	**100,0**
nursery (12960, 8, 5)	43,3	**99,3±0,0**	-	**99,3**^{0}	$98,1^{-5}$	$99,2^{-4}$	97,1	99,0	99,0
pendigits (10992, 16, 10)	1,2	**99,4±0,0**	-	**99,4**^{0}	**99,4**^{0}	$97,4^{-5}$	96,7	98,2	98,8
pima (768, 8, 2)	34,3	74,7±0,9	$65,2^{-5}$	$74,4^{0}$	$72,2^{-5}$	$72,7^{-5}$	73,0	**76,8**	73,2
primary (336, 15, 21)	75,9	31,7±0,8	$32,4^{+1}$	**40,3**$^{+5}$	$33,5^{+4}$	$31,6^{0}$	-	-	-
satimage (6435, 36, 6)	3,7	91,3±0,1	-	$91,3^{0}$	**91,4**$^{+2}$	$87,7^{-5}$	86,7	88,5	90,8
segment (2310, 19, 7)	1,7	97,4±0,1	$45,3^{-5}$	**97,5**$^{+2}$	$97,3^{-2}$	$92,1^{-5}$	97,3	95,0	96,6
shuttle (58000, 9, 7)	1,3	**99,9±0,0**	-	**99,9**^{0}	**99,9**^{0}	$99,8^{-5}$	99,6	97,0	99,7
solar-flare (1066, 10, 8)	70,9	81,2±0,3	$81,4^{+1}$	$82,7^{+5}$	$78,1^{-5}$	$81,7^{+5}$	82,7	**83,5**	**83,5**
splice (3186, 60, 3)	17,3	93,9±0,2	-	$93,9^{0}$	$94,0^{0}$	**94,6**$^{+5}$	94,2	69,7	69,7
wine (178, 13, 3)	10,1	**97,2±0,6**	$40,1^{-5}$	**97,2**^{0}	$96,9^{0}$	$94,5^{-5}$	93,3	95,6	95,5
yeast (1484, 8, 10)	23,0	**59,8±0,6**	$45,9^{-5}$	$58,1^{-5}$	$54,9^{-5}$	$59,1^{-4}$	56,1	**59,8**	54,6
Total Average		**88,7±0,4**	64,3	**88,7**	87,8	87,3	86,6	86,1	87,1

from the paper [8]. The remaining algorithms were tested on a 800MHz PentiumIII PC, with 512M bytes of RAM. The algorithm *RIA* is time expensive so it was tested only for smaller data sets. The results were obtained by performing 10-fold cross-validation 10 times for each data set. All implemented algorithms: *RIONA, RIA, ONN, 3-NN* and *RIONA(B)* were tested with exactly the same folds and the significance of difference between algorithms was estimated using one-tailed paired t test.[1] SVDM metric and the optimal neighbourhood were computed from a training set independently for each run in a cross-validation test.

[1] The result of a single cross-validation test was the accuracy averaged over all 10 folds and the final average accuracy and the confidence level for difference between *RIONA* and the corresponding algorithm were computed from 10 repeats of the cross-validation test (for *census-income* and *shuttle* only 4 repeats).

The total average accuracy was computed over all data sets except *breast, bupa-liver* and *primary* (for *RIA* it was computed only over the data sets that are given the accuracy).

For all data sets the presented results were obtained for the metric described in Section 3.2 and $NormNStrength$ measure for conflict resolving (see Section 4). Although during the preliminary experiments we tried other types of a metric, no one appeared better then the presented one in terms of accuracy on a range of problem domains. We also tried to omit normalisation factor in the measure $NormNStrength$ what gave almost identical results. The optimal size of a neighbourhood was searched during the process of learning on the basis of the training examples. From the time complexity perspective it was important to limit searching for the optimal k to a small fixed range of possible values from 1 to k_{max} in such a way that sorting and consistency checking of k_{max} nearest neighbours were efficient. Since the values k_{max} optimal in this sense are the values close to the square root of the training set size (see Section 5.2) we set $k_{max} = 200$ (it is close to the square root of the size of the largest domains). In the next subsection we examine the significance of this setting.

In Table 1 one can see that significant differences in accuracy between *RIONA* and *ONN* (*k-NN* with selection of the optimal neighbourhood) occurred mostly for smaller data sets (*breast*, *bupa-liver*, *chess*, *primary*, *solar-flare* and *yeast*). The only difference between *RIONA* and *ONN* is the operation of consistency checking. In order to explain the similarity of results we checked what part of the k-neighbourhood for the optimal k is eliminated by the operation of consistency-checking and found that only for the domains *breast*, *primary* and *solar-flare* the fraction of eliminated nearest neighbours was significant. For other domains the number of consistent objects from the optimal neighbourhood in *RIONA* algorithm is close to the number of all objects from the optimal neighbourhood of *k-NN* algorithm. Therefore the differences in classification accuracy are small. These observations suggest that the operation of consistency checking in *RIONA* is not very significant and it should be considered to be more restrictive.

On the other hand, the accuracy of *RIONA* and *ONN* is comparable or better than well-known classifiers, in particular, their accuracy is generally better than the accuracy of *RIA* and *3-NN*. It suggests the conclusion that *RIONA* and *ONN* may replace successfully both the rule-based algorithm using all minimal rules and a *k-NN* with a fixed k. It also proves that using a properly selected subset of rules in rule-based systems gives better results than using all minimal rules. The range of tested data sets indicates that the presented algorithms work well for domains with both numerical and symbolic attributes. In particular, it works well for numerical attributes without preprocessing.

5.1 Further Study

In this section we describe more experiments and conclusions that can help us to understand important aspects of *RIONA*.

First, we performed the experiments that helped us to compare two types of a neighbourhood: thr radial $B(tst, R)$ and the k-best $S(tst, k)$. For each data

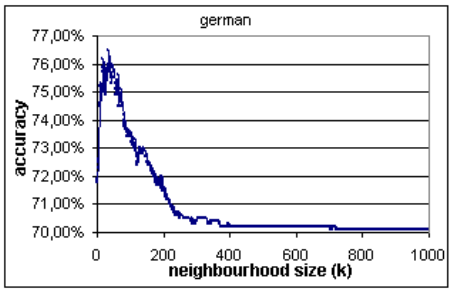

Fig. 1. Accuracy for *german*

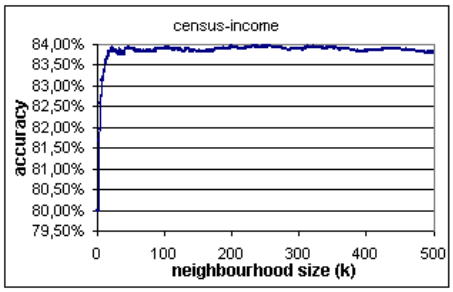

Fig. 2. Accuracy for *census-income*

set we estimated the optimal value of the radius R and the optimal value of k from a training set and compared classification accuracy for both types of a neighbourhood. Looking at the third and the seventh columns in Table 1 one can see that the accuracy of the algorithm for the neighbourhood $B(tst, R)$ is significantly worse than $S(tst, k)$ on 14 domains (with the confidence level -4, -5) and significantly better on 3 domains (with the confidence level +4, +5). Therefore in further experiments we focused our attention on the neighbourhood $S(tst, k)$.

The setting $k_{max} = 200$ preserved the efficiency of *RIONA* but the interesting question was how significantly this setting influenced the classification results. Please note that the maximal possible value k is just the size of a training set. In order to answer this question the following experiment was performed: for the smaller sets (less than 4000 objects) the classification accuracy was measured for all possible values of k and for the greater sets the maximal value k was set to $k_{max} = 500$ (for the set *nursery* we made the exception $k_{max} = 1000$). The classification accuracy was measured for the leave-one-out method applied to the whole sets. Figures 1, 2 present the dependence of classification accuracy on the value of k for exemplary domains.

For most data sets we observed that while increasing k beyond a certain small value the classification accuracy is falling down (see Figure 1). In particular, while comparing the third and the fourth column in Table 1, one can see that for most data sets the results for the total neighbourhood are significantly worse than the results for the neighbourhood found by the algorithm *RIONA*. For the remaining data sets (*breast, census-income, nursery, primary, solar-flare*) the accuracy becomes stable beyond a certain value k (see Figure 2).

For the former group we examined the neighbourhood size (the value of k) for which the maximum accuracy was obtained. In the latter case we examined both the value of k beyond which the accuracy remains stable and the fluctuations in accuracy while increasing k. For most domains the optimal value of k appeared to be much less than 200. On the other hand, for the domains where the optimal k was greater (*australian, census-income* and *nursery*) the loss in accuracy related to this setting was insignificant: it remained within the range of 0,15%. Moreover,

Table 2. Single object test time (in seconds) for *RIONA*, *RIA* and *ONN*

Domain	t_{RIONA}	t_{RIA}	t_{ONN}	Domain	t_{RIONA}	t_{RIA}	t_{ONN}
australian	0,026	0,087	0,022	breast	0,016	0,021	0,014
breast-wis	0,032	0,063	0,017	bupa-liver	0,009	0,016	0,006
census	0,572	$> 5,0$	0,568	chess	0,130	0,891	0,126
german	0,047	0,188	0,042	glass	0,010	0,012	0,006
heart	0,019	0,024	0,014	iris	0,003	0,006	0,003
letter	0,236	$> 5,0$	0,224	lymph	0,017	0,019	0,014
mushroom	0,223	$> 5,0$	0,219	nursery	0,169	$> 5,0$	0,167
pendigits	0,133	$> 5,0$	0,130	pima	0,013	0,055	0,010
primary-tumor	0,018	0,028	0,018	satimage	0,174	$> 5,0$	0,169
segment	0,046	0,557	0,042	shuttle	0,378	$> 5,0$	0,376
solar-flare	0,025	0,082	0,023	splice	0,405	3,194	0,393
wine	0,010	0,891	0,007	yeast	0,017	0,104	0,014

the accuracy became stable for values of k also much lower than 200. Therefore we could conclude that the setting $k_{max} = 200$ preserved good time complexity properties and did not change the results significantly for tested data sets.

For data sets split originally into a training and a testing set (*splice, satimage, pendigits, letter, census-income, shuttle*) we performed the experiments to compare the accuracy for two cases: when the value k was estimated either from a training set or from a test set (the optimal k). Experiments showed that for *pendigits* accuracy obtained by *RIONA* differs by about half percent from the accuracy with an optimal number k and for the other domains the difference remains in the range of 0.2%. It means that the used algorithm finds almost optimal number k in terms of obtained accuracy.

Analogical experiments were done for the neighbourhood $B(tst, R)$ and we observed that after the value R exceeded a constant R_{max} (where R_{max} was relatively small in comparison to the maximal possible value of R) the accuracy either became worse or did not improve significantly. This suggests the similar conclusion, i.e. the best accuracy is obtained for a small radius.

5.2 Time Complexity of RIONA

First, the learning algorithm performs two phases for each training object. In the first phase it selects k_{max} nearest objects among $n = |trnSet|$ objects. On average it is done in the linear time. In the second phase the algorithm sorts all k_{max} selected objects and checks consistency among them. It takes $O(k_{max}^2)$. Finally, for the whole training set the algorithm computes leave-one-out accuracy for each $1 \leq k \leq k_{max}$, which takes $O(nk_{max})$. Summing up, the average complexity of the learning algorithm is $O(n(n + k_{max}^2))$. In practice the component $O(n^2)$ is dominant.

Testing is analogical to learning. The classification algorithm finds k_{opt} nearest examples and then checks consistency among them. Since $k_{opt} \leq k_{max}$, the complexity is $O(n + k_{max}^2)$ for a single test object and the total average com-

plexity of the testing algorithm is $O(m(n + k_{max}^2))$ where m is a number of test objects. In Table 2 one can see that for all the presented data sets the average time of classification for a single object is less than 0.6 s. Moreover, for larger data sets it is comparable with a single object test time in the algorithm *ONN* and is much shorter than a single test object time in the algorithm *RIA*.

In case when the number of test objects is approximately equal to the number of training objects, taking into account both the learning and the classification phase, the average time complexity of *RIONA* is in practise $O(n^2)$, while the average time complexity of *RIA* is $O(n^3)$ what is quite a significant acceleration.

6 Conclusions and Future Research

The research reported in the paper attempts to bring together the features of rule induction and instance-based learning in a single algorithm. As the empirical results indicate the presented algorithm obtained the accuracy comparable to the well-known systems such as: *3-NN*, *C5.0*, *DeEPs* and *DeEPsNN*. The experiments show that the choice of a metric is very important for classification accuracy of the algorithm. The combination of the normalised Manhattan metric for numerical attributes and SVDM metric for symbolic attributes proved to be very successful. It did not require discretisation for numerical attributes.

We have compared two types of a neighbourhood: the k-nearest neighbours $S(tst, k)$ and the ball $B(tst, R)$. The former type of a neighbourhood gave generally better results, although the latter seemed more natural. This may suggest that the topology of the space induced by the used metric is rather complex.

We found that the appropriate choice of the neighbourhood size is also an important factor for classification accuracy. It appeared that for all domain problems the optimal accuracy is obtained for a small neighbourhood (a small number of nearest neighbours k in S or a small radius R in B neighbourhood). This leads us to the conclusion that generally it is enough to consider only a small neighbourhood instead of the maximal neighbourhood related to the whole training set. This is interesting from the classification perspective, because it suggests that usually only a small number of training examples is relevant for accurate classification. It also illustrates the empirical fact that while using rule-based classifiers one can obtain better results by rejecting some rules instead of using all minimal rules like the algorithm *RIA* does. We propose an approach to use only the rules that are built on the basis of a neighbourhood of the test case.

The fact mentioned above is also the key idea that allowed us to make the original algorithm *RIA* efficient without loss in classification accuracy. In practice the complexity of learning and classification is only squarely and linearly dependent on the size of a learning sample respectively. Although a great effort was put into accelerating the algorithm, we think that further acceleration is possible, for instance by more specialised data structures and an approximate choice of nearest examples (see e.g. [10]).

The facts that *RIONA* and *ONN* algorithms have similar classification accuracy and the fraction of objects eliminated by the consistency checking operation

is very small indicate that this operation has rather small influence on the accuracy of the algorithm. It suggests that the *k-NN* component remains a dominant element of *RIONA* and shows that either the construction of local rules should be more general or the operation of consistency checking should be more restrictive.

In *RIONA* the selection of the optimal value of k is performed globally. One possible extension of this approach is to apply a local method to searching for the appropriate value of k (see e.g. [12]).

The interesting topic is the dependence of the average number of training examples on the distance to a test case. Empirically it was noticed that the dependence was close to linear, what seemed surprising to us.

Acknowledgements The authors are very grateful to professor Andrzej Skowron for his useful remarks on this presentation. This work was supported by the grants 8 T11C 009 19 and 8 T11C 025 19 from the Polish National Committee for Scientific Research.

References

1. Bazan, J.G. (1998). *Discovery of decision rules by matching new objects against data tables.* In: L. Polkowski, A. Skowron (eds.), Proceedings of the First International Conference on Rough Sets and Current Trends in Computing (RSCTC-98), pages 521-528, Warsaw, Poland.
2. Blake, C.L., Merz, C.J. (1998). *UCI Repository of machine learning databases* [http://www.ics.uci.edu/~mlearn/MLRepository.html], Department of Information and Computer Science, Irvine, CA: University of California.
3. Cost, S. and Salzberg, S. (1993). *A weighted nearest neighbor algorithm for learning with symbolic features.* Machine Learning, 10, pages 57-78.
4. Domingos, P. (1996). *Unifying instance-based and rule-based induction.* Machine Learning, 24(2), pages 141-168.
5. Duda, R.O. and Hart, P.E. (1973). *Pattern classification and scene analysis.* New York, NY: Wiley.
6. Golding, A.R., Rosenbloom, P.S. (1991). *Improving rule-based systems through case-based reasoning.* Proceedings of AAAI-91, pages 22-27, Anaheim, CA.
7. Li, J., Ramamohanarao, K. and Dong, G. (2001). *Combining the strength of pattern frequency and distance for classification.* The Fifth Pacific-Asia Conference On Knowledge Discovery and Data Mining, pages 455-466, Hong Kong.
8. Li, J., Dong, G., Ramamohanaro, K. and Wong, L. (2001). *DeEPs: A new instance-based discovery and classification system.* [http://sdmc.krdl.org.sg:8080/~limsoon/limsoonpapers.html], School of Computing, National University of Singapore.
9. Michalski, R.S., Mozetic, I., Hong, J. and Lavrac, N. (1986) *The Multi-Purpose Incremental Learning System AQ15 and its Testing to Three Medical Domains.* Proceedings of AAAI-86, pages 1041-1045, San Mateo: Morgan Kaufmann.
10. Mitchell T.M. (1997). *Machine learning.* Portland: McGraw-Hill.
11. Schaffer, C. (1994). A conservation law for generalisation performance. Proceedings of the Twelfth International Conference on Machine Learning, pages 259-265, New Brunswick, NJ: Morgan Kaufmann.
12. Wettschereck, D. (1994). *A study of Distance-Based Machine Learning Algorithms.* Doctor of Philosophy dissertation in Computer Science, Oregon State University.