2011 11th IEEE International Conference on Data Mining

Classifying Categorical Data by Rule-based Neighbors

Jiabing WANG[1], Pei ZHANG[1]
[1]School of Computer Science and Engineering
South China University of Technology
Guangzhou, China
jbwang@scut.edu.cn, z.pei02@mail.scut.edu.cn

Guihua WEN[1, 2], Jia WEI[1]
[2]State Key Laboratory of Brain and Cognitive Science
Chinese Academy of Science
Beijing, China
{crghwen, csjwei}@scut.edu.cn

***Abstract*—A new learning algorithm for categorical data, named CRN (Classification by Rule-based Neighbors) is proposed in this paper. CRN is a nonmetric and parameter-free classifier, and can be regarded as a hybrid of rule induction and instance-based learning. Based on a new measure of attributes quality and the separate-and-conquer strategy, CRN learns a collection of feature sets such that for each pair of instances belonging to different classes, there is a feature set on which the two instances disagree. For an unlabeled instance *I* and a labeled instance *I′*, *I′* is a neighbor of *I* if and only if they agree on all attributes of a feature set. Then, CRN classifies an unlabeled instance *I* based on *I*'s neighbors on those learned feature sets. To validate the performance of CRN, CRN is compared with six state-of-the-art classifiers on twenty-four datasets. Experimental results demonstrate that although the underlying idea of CRN is simple, the predictive accuracy of CRN is comparable to or better than that of the state-of-the-art classifiers on most datasets.**

Keywords-classification; categorical data; feature selection; rule-based neighbors

I. INTRODUCTION

Instance-based learning [1] is an important learning paradigm. The *k*-Nearest-Neighbor (abbr. *k*-NN) [2] is a representative instance-based classifier that assigns an unlabeled instance with the most common class among its *k* nearest neighbors. Due to its simplicity and effectiveness, *k*-NN classifiers have been widely employed in pattern classification field.

Most of the instance-based classifiers use a given metric to measure the similarity between the unlabeled instance and its neighbors. When attributes are numerical, the normalized Euclidean distance is a natural metric to measure the similarity between instances. However, there may not exist some natural notion of metric for many applications. In this case, many instance-based classifiers that are designed to handle numerical attributes will be confronted with difficulty and typically use much simpler metrics to measure the distance between values of categorical attributes. Although those simpler metrics perform well in some cases, they may fail to capture the inherent complexity of the problem domains, and as a result may perform badly [3].

One issue of instance-based classifiers is the choice of parameter *k*, i.e., the number of neighbors, which is often chosen empirically by cross-validation or domain experts in practice. Apart from the parameter *k*, many *k*-NN classifiers need options and other parameters. In order to make those classifiers perform best, those parameters must be fine-tuned. As the same of choice of *k*, most parameters can only be empirically obtained by cross-validation or a similar method, which is usually a tedious work. Another issue of instance-based classifiers is that they are very sensitive to irrelevant attributes. If a dataset contains a large number of irrelevant attributes, the contributions of these attributes to the global distance will dramatically decrease the performance of instance-based classifiers [4].

Attempt to deal with the drawbacks of instance-based learning as mentioned above, a new instance-based learning algorithm for categorical data, named CRN (Classification by Rule-based Neighbors), is proposed in this paper. Unlike conventional instance-based classifiers, we treat "neighbor" in a rule-like way which is essentially different from distance metric view: for a test instance *I* and a labeled instance *I′*, *I′* is a neighbor of *I* if and only if they agree on a feature set (a subset of attributes). As a result, CRN does not use any metric to measure the similarity between a test instance and its neighbors. Since there are exponential-size subsets of attributes, a new measure of attributes quality is proposed in order to learn "high quality" feature sets. Based on the new quality measure of attributes and the separate-and-conquer strategy commonly used in rule induction [5], a collection of feature sets are learned. Then, an unlabeled test instance is classified by neighbors defined on those feature sets.

The rest of this paper is organized as follows. Section II presents the CRN algorithm including the definition of the measure of attributes quality, the training algorithm and the classification algorithm. Section III reports the experimental results. We conclude the paper in Section IV.

We use the following notations in the rest of the paper:

D: the training set.

D_c: the subset of D consisting of instances with class c.

n: the number of instances in D.

m: the number of attributes, and the attribute's name is $A_1, A_2, \ldots,$ and A_m respectively.

T: the number of total class labels, and the class label name is 1, 2, …, and T respectively.

II. THE CRN ALGORITHM

We propose the CRN algorithm based on a simple idea: if two instances belong to different classes, there is at least one attribute on which two instances disagree. So, the core of

CRN is to learn a collection of feature sets such that for each pair of instances belonging to different classes, there is a feature set on which two instances disagree.

There are two phases in CRN, i.e., the training phase and the classification phase. In the training phase, CRN learns a collection S of feature sets based on a new measure of attributes quality. In the classification phase, CRN classifies an unlabeled instance using those feature sets.

A. *The Measure of Attributes Quality*

Definition 2.1. Given an instance $I \in D_c$, we say that an instance $I' \in D - D_c$ *is discriminated from* I *by an attribute* A if $I_A \neq I'_A$, where "–" denotes the set subtraction. Given a feature set V and an instance $I \in D_c$, we say that *all instances in* $D - D_c$ *are discriminated from* I *by* V if for each instance $I' \in D - D_c$, there is an attribute $A \in V$ such that $I_A \neq I'_A$.

Definition 2.2. Given an instance $I \in D_c$, an attribute A and a subset $D_c^{\sim} \subseteq D - D_c$, we define $C(A, D_c^{\sim})$ to be the number of instances in $D_c^{\sim}$ that are discriminated from I by A, that is, the number of instances which have different class from c and different values from I on A.

Given an instance $I \in D_c$, CRN learns a feature set by iteratively selecting an attribute and adding it into the feature set V until all instances in $D - D_c$ are discriminated from I by V. The key issue is: which attribute should be selected in each iteration, about which we have the following two considerations. First, the selected attribute should have high classification ability; second, in order to avoid overfitting and for the sake of comprehensibility, the learned feature set should be small.

In decision tree and rule induction learning paradigms, the quality of an attribute is usually measured in terms of the *purity* of classes of training instances. A set of instances is *pure* if all instances belong to the same class and maximally *impure* if the proportion of instances for each class is uniform. Two well-known and commonly used impurity measures are *entropy*, used in ID3 decision tree classifier [6] and its successor C4.5 [7], and the *gini-index*, used in CART system [8]. In this paper, we will define the measure of attributes quality based on entropy and split information. We now first review the concept of entropy.

For an attribute A having p distinct values $\{1, 2, \ldots, p\}$, let D_c^i denote the set of instances in D whose attribute A takes the value i and are labeled class c, and D_*^i be the union of D_c^i over c, $i = 1, 2, \ldots, p$, $c = 1, 2, \ldots, T$. The *entropy* of the attribute A on D is computed as (1) [6],

$$E(A,D) = -\sum_{i=1}^{p} \frac{|D_*^i|}{|D|} \sum_{c=1}^{T} \left(\frac{|D_c^i|}{|D_*^i|} \log_2 \left(\frac{|D_c^i|}{|D_*^i|}\right)\right). \qquad (1)$$

Note that in (1), the logarithm of zero is defined to be zero. The entropy of an attribute A provides a measure of impurity for the subsets D_*^i of D, $i = 1, 2, \ldots, p$. Especially, if $E(A, D)$ equals zero, then D_*^i is pure for $i = 1, 2, \ldots, p$, and the attribute A can correctly classify all instances in D.

On the other hand, given an instance $I \in D_c$, the problem "*learning the smallest feature set* V *such that all instances in* $D - D_c$ *are discriminated from* I *by* V" is the classical set cover problem and is NP-hard [9]. The Johnson's greedy strategy [10] is a well-known approximate algorithm to solve the set cover problem: in each iteration, the attribute A is selected into V such that $C(A, D_c^{\sim})$ is maximum among unselected attributes. Johnson algorithm achieves the approximation ratio $\ln|D_c^{\sim}|$ and it is hard to achieve a $(1 - \varepsilon)\ln|D_c^{\sim}|$ approximation ratio for any $\varepsilon > 0$ [11].

Based on the above discussion, when learning a feature set V for an instance $I \in D_c$, in each iteration we should select the attribute A such that the entropy of A is *minimum* and $C(A, D_c^{\sim})$ is *maximum* among all unselected attributes. However, it is unlikely that the two maxima can be simultaneously achieved. Therefore, to make a trade-off between the impurities of the partitions of D divided by an attribute and the cardinality of a feature set, it is natural to propose the measure of attributes quality as described in Definition 2.3. Now, when learning a feature set V for an instance $I \in D_c$, the attribute A whose $Q(A, D_c^{\sim}, D)$ value is minimum among all unselected attributes is selected into V in the current iteration.

Definition 2.3. Given an instance $I \in D_c$, and a subset $D_c^{\sim} \subseteq D - D_c$, $c = 1, 2, \ldots, T$, the quality of an attribute A with respect to $D_c^{\sim}$ is defined as (2),

$$Q(A, D_c^{\sim}, D) = \begin{cases} +\infty & if\ C(A, D_c^{\sim}) = 0 \\ \dfrac{E(A,D)}{C(A, D_c^{\sim})} \times SI(A,D) & else \end{cases}, \qquad (2)$$

where $SI(A, D)$ is computed as (3), and the meaning of p and D_*^i in (3) can be found in the description about (1),

$$SI(A,D) = -\sum_{i=1}^{p} \frac{|D_*^i|}{|D|} \log_2 \frac{|D_*^i|}{|D|}. \qquad (3)$$

Note that in Definition 2.3, $E(A, D)$ and $SI(A, D)$ are defined on D and are constant in whole training phase.

The term $SI(A, D)$ in (2), proposed by Quinlan in C4.5 [7], is called the *split information* and used to overcome the bias that the terms $E(A, D)$ and $C(A, D_c^{\sim})$ have: they may be biased by the attributes that have a great number of values.

B. *The Training Phase*

In the training phase, CRN learns a collection S of feature sets such that for each instance $I \in D$, there is a feature set $V \in S$ that discriminates I from all instances labeled with different classes from I's. The feature sets are learned using the separate-and-conquer strategy [5], that is, whenever a feature set is learned for an instance I, it is needless to learn feature sets for the *neighbors* (Definition 2.4) of I.

Definition 2.4. Given a training set D, a feature set V, a labeled instance $I' \in D_c$, $c = 1, 2, \ldots, T$, and an instance I, we say that I' *is the neighbor of* I *on* V *with class* c if and only if for each attribute $A \in V$, I and I' agree on A: $I_A = I'_A$.

```
Procedure Learning_Feature_Sets(D)
Input: the training set $D$.
Output: the collection $S$ of learned feature sets.
for each attribute $A_i$, $i = 1, 2, \ldots, m$, computing the entropy $E(A_i, D)$ using the formula (1) and the split information $SI(A_i, D)$ using the formula (3);
for each attribute $A_i$, $i = 1, 2, \ldots, m$, computing the product, denoted as $ES(A_i)$, of $E(A_i, D)$ and $SI(A_i, D)$;
$S \leftarrow \Phi$;   %initialized as empty collection.
for $c = 1$ to $T$
    $Temp \leftarrow D_c$;
    $D_c^{\sim} \leftarrow D - D_c$;
    while($Temp \neq \Phi$)
        $I \leftarrow$ an instance in $Temp$;
        for each attribute $A_i$, $i = 1, 2, \ldots, m$, computing the $Count(A_i, D_c^{\sim})$;
        $V \leftarrow Learning\_One\_Feature\_Set(I, ES, Count, c, D_c^{\sim})$;
        if($V \notin S$)
            $S \leftarrow S \cup \{V\}$;
        Endif
        $Temp \leftarrow Temp$ – {the neighbors of $I$ on $V$};
    endwhile
endfor
return $S$;

Subroutine Learning_One_Feature_Set(I, ES, Count, c, $D_c^{\sim}$)
Output: a feature set $V$.
$V \leftarrow \Phi$;
$Attribute\_Set \leftarrow \{A_1, A_2, \ldots, A_m\}$;
for each $A \in Attribute\_Set$, computing $Q(A, D_c^{\sim}, D)$ using the formula (2);
while($D_c^{\sim} \neq \Phi$ and $Attribute\_Set \neq \Phi$)
    $A_{min} \leftarrow \arg\min_{A \in Attribute\_Set} Q(A, D_c^{\sim}, D)$;
    $V \leftarrow V \cup \{A_{min}\}$;
    $Attribute\_Set \leftarrow Attribute\_Set - \{A_{min}\}$;
    $D_c^{\sim} \leftarrow D_c^{\sim}$ – {instances discriminated from $I$ by $A_{min}$};
    for each $A \in Attribute\_Set$, computing $Count(A, D_c^{\sim})$;
    for each $A \in Attribute\_Set$, computing $Q(A, D_c^{\sim}, D)$ using the formula (2);
endwhile
return $V$;
```

Figure 1. The training algorithm.

Now the training algorithm, named *Learning_Feature_Sets*, is described in Fig. 1. First, the entropy and the split information of each attribute are computed using (1) and (3) respectively. Then for each class c, $c = 1, 2, \ldots, T$, CRN learns a collection of feature sets by calling the subroutine *Learning_One_Feature_Set*. Specifically, after initializing *Temp* as the subset D_c of D and $D_c^{\sim}$ as $D - D_c$, then repeating the following procedure until *Temp* becomes empty: picking an instance I from *Temp* and computing $C(A, D_c^{\sim})$ for each attribute A; calling the subroutine *Learning_One_Feature_Set* to learn a feature set V; adding V to S if V does not belong to S; deleting all neighbors of I on V from *Temp*.

The subroutine *Learning_One_Feature_Set* learns a feature set V that discriminates all instances in $D_c^{\sim}$ from I. After initializing *Attribute_Set* as the set of all attributes and computing the quality of each attribute in *Attribute_Set* with respect to $D_c^{\sim}$ using (2), *Learning_One_Feature_Set* learns a feature set using the following greedy strategy: selecting the attribute A_{min} such that the quality of A_{min} is minimum among *Attribute_Set*; adding A_{min} into the feature set V and removing the attribute A_{min} from *Attribute_Set*; removing the instances from $D_c^{\sim}$ that are discriminated from I by A_{min}; recomputing the number of instances in $D_c^{\sim}$ discriminated from I by A and the quality of A using (2) for each attribute $A \in$ *Attribute_Set*; and repeating the procedure until $D_c^{\sim}$ or *Attribute_Set* becomes empty.

C. *The Classification Phase*

In this section, we use $N_c(V)$ to denote the number of neighbors of an instance I on a feature set V with class c.

Definition 2.5. Let *SUM* be the sum of $N_c(V)$ over c, i.e., the number of all neighbors of I on V. Define the *impurity of neighbors of I on V* as (4).

$$IP(V) = \begin{cases} -\sum_{c=1}^{T} \frac{N_c(V)}{SUM} \log_2 \frac{N_c(V)}{SUM}, & if\ SUM > 0 \\ +\infty & else \end{cases} \quad (4)$$

Similar to (1), the logarithm of zero is also defined to be zero in (4). It is obvious that if the impurity of neighbors of an instance I on a feature set V is zero, then all neighbors of I on V belong to the same class.

Definition 2.6. Define *the candidate label of I on V* as (5),

$$CL(V) = \arg\max_{c \in \{1:T\}} N_c(V), \quad (5)$$

that is, the candidate label $CL(V)$ is the class label with which the number of neighbors of I on V is maximum among all class labels. The corresponding $N_{CL(V)}(V)$ is called *the dominant neighbor count of I on V*.

To predict the class of an unlabeled instance I, we have the following three hypotheses. First, from (4), it is obvious that if the impurity of neighbors of I on a feature set V is zero, then all neighbors of I on V belong to the same class. Therefore, we have the first hypothesis: *the less the impurity of neighbors of I on a feature set V is, the more probable that I belongs to the candidate label of I on V*. Second, according to the principle of Occam's razor [12], we have the second hypothesis: *the smaller a feature set V is, the more probable that I belongs to the candidate label of I on V*. Finally, the third hypothesis is: *the greater the dominant neighbor count of I on V is, the more probable that I belongs to the candidate label of I on V*.

Therefore, CRN predicts the class of I using the impurity of neighbors of I on V together with the cardinality of V and the dominant neighbor count of I on V.

```
Procedure Classification(D, S, I)
Input: the training set D, the collection S of feature sets returned by
        the procedure Learning_Feature_Sets, and an unlabeled
        instance I.
Output: the class label L of I.
for each feature set $V \in S$
    counting the number of neighbors of I on V with class c, $c = 1, 2, \ldots, T$;
    computing the impurity $IP(V)$ of neighbors of I on V using the formula (4);
    if(the number of neighbors of I on V does not equal zero)
        finding the candidate label $CL(V)$ and the corresponding
        dominant neighbor count $N_{CL(V)}(V)$ of I on V using the formula (5);
    endif
endfor
$IP_{min} \leftarrow \min_{V \in S} IP(V)$;
if($IP_{min} \neq +\infty$)
    for each feature set $V \in S$, computing the $Priority(V)$ using the formula (6);
    $Priority_{max} \leftarrow \max_{V \in S} Priority(V)$;
    $VS \leftarrow \{V \mid Priority(V) = Priority_{max}\}$;
    $CLS \leftarrow \{CL(V) \mid V \in VS\}$;
    $L \leftarrow \arg\min_{c \in CLS} |D_c|$;
    return L;
else
    $L \leftarrow \arg\max_{c \in \{1,2,\cdots,T\}} |D_c|$;
return L;
endif
```

Figure 2. The classification algorithm.

Let IP_{min} be the minimum impurity among all feature sets. If $IP_{min} \neq +\infty$, we define the *priority of candidate label of I on V* as (6),

$$Priority(V) = \begin{cases} \dfrac{N_{CL(V)}(V)}{|V|} & if\ IP(V) = IP_{min} \\ 0 & else \end{cases}. \quad (6)$$

Now, the algorithm in the classification phase is given in Fig. 2. For each feature set $V \in S$, CRN counts the number of neighbors of *I* on *V* with class *c* for $c = 1, 2, \ldots, T$, and computes the impurity $IP(V)$ of neighbors of *I* on *V* using (4). If the number of neighbors of *I* on *V* does not equal zero, then finds the candidate label $CL(V)$ of *I* on *V*, and the corresponding dominant neighbor count $N_{CL(V)}(V)$ using (5).

If $IP_{min} \neq +\infty$, compute $Priority(V)$ using (6) for each feature set *V* and let $Priority_{max}$ be the maximum priority among priorities of candidate labels of *I* on all feature sets. Then, the class label of *I* is decided by all candidate labels of *I* on feature sets *V* such that $Priority(V) = Priority_{max}$. If there is only one feature set *V* such that $Priority(V) = Priority_{max}$, the prediction is simple: return the candidate label $CL(V)$ of *I* on *V* as the class label of *I*. However, for an instance *I*, there may exist a set *VS* of feature sets such that for each $V \in VS$, $Priority(V) = Priority_{max}$ and the candidate labels of *I* on *VS* are different. To break the ties, let *CLS* be the set of candidate labels of *I* on *VS*, CRN will return *L* as the class label of *I* such that $|D_L|$ is minimum among *CLS*.

For some instance *I*, if the number of neighbors of *I* is zero, i.e., $IP_{min} = +\infty$, CRN returns the most common class among the training set *D*.

The worst-case time complexity with the number of instances in the training phase is $O(n^2)$, and the time for classifying an instance is $O(n|S||V|)$, and $|V|$ is usually a small number. The space complexity in both phases is $O(nm)$, which is optimum since $O(nm)$ space is required merely to store the training set *D*.

III. THE EXPERIMENTAL RESULTS

The goal of our experiments is to answer the following questions:

(1) How does the predictive accuracy of CRN compare to that of popular and state-of-the-art classifiers?

(2) Is CRN sensitive to irrelevant attributes? Is CRN biased by attributes with a great number of values?

(3) Is CRN efficient compare with state-of-the-art classifiers?

(4) How big are the number of feature sets and the number of attributes in a feature set?

However, considering the space limitation, here we only give the answers to the first two questions.

TABLE I. THE INFORMATION OF BENCHMARK DATASETS

#	Dataset name	Size	#Class	#Attribute	#Majority	#Minority
1	Balance	625	3	4	288	49
2	Breast-w	699	2	9	458	241
3	Extended_Monks-1	1000	2	100	554	446
4	Extended_Monks-2	1000	2	100	649	351
5	Extended_Monks-3	1000	2	100	627	373
6	Kr-vs-kp	3196	2	36	1669	1527
7	Led-24	5000	10	24	542	452
8	Lymphography	148	4	18	81	2
9	Monks-1	Train: 124	2	6	62	62
		Test: 432			216	216
10	Monks-2	Train: 169	2	6	105	64
		Test: 432			290	142
11	Monks-3	Train: 122	2	6	62	60
		Test: 432			228	204
12	Mushroom	8124	2	22	4208	3916
13	Nursery	Train: 10344	5	8	3454	1
		Test: 2616			901	1
14	Promoters	106	2	57	53	53
15	RDG10000-200	10000	2	200	5975	4025
16	RDG10000-200-50	10000	2	200	6831	3169
17	RDG3000-1000	3000	3	1000	1933	240
18	RDG3000-1000-500	3000	3	1000	1395	387
19	Soybean	47	4	35	17	10
20	Spect	Train: 80	2	22	40	40
		Test: 187			172	15
21	Splice	3175	3	60	1648	762
22	Tic-tac-toe	958	2	9	626	332
23	Voting	435	2	16	267	168
24	Zoo	101	7	17	20	4

TABLE II. THE MEAN PREDICTIVE ACCURACY AND THE STANDARD DEVIATION ON UCI DATASETS

Dataset	CRN	IBL(k=11)	C4.5	LMT	RF	PART	RIPPER
Balance	82.93±0.83	84.55±0.88	64.14±1.20	**93.30±0.32**	78.83±1.32	76.96±1.40	73.25±1.47
Breast-w	95.58±0.33	94.05±0.17	94.39±0.48	95.45±0.64	**96.52±0.20**	93.85±0.53	94.01±0.61
kr-vs-kp	99.06±0.13	95.06±0.36	99.44±0.14	**99.75±0.08**	99.28±0.13	99.06±0.17	99.25±0.17
Led-24	70.73±0.32	70.12±0.24	72.58±0.23	**73.62±0.50**	72.74±0.30	64.28±0.20	70.98±0.19
Lymphography	**84.46±0.00**	81.08±0.00	79.05±0.00	81.76±0.00	83.78±0.00	**84.46±0.00**	78.38±0.00
Monks-1	92.52±0.00	76.39±0.00	75.69±0.00	75.69±0.00	93.75±0.00	**100.0±0.00**	61.11±0.00
Monks-2	**79.75±0.00**	66.20±0.00	65.05±0.00	62.50±0.00	68.75±0.00	70.14±0.00	62.50±0.00
Monks-3	96.06±0.00	92.82±0.00	**97.22±0.00**	**97.22±0.00**	96.53±0.00	**97.22±0.00**	86.11±0.00
Mushroom	**100.0±0.00**	**100.0±0.00**	**100.0±0.00**	**100.0±0.00**	**100.0±0.00**	**100.0±0.00**	**100.0±0.00**
Nursery	98.85±0.00	98.59±0.00	96.94±0.00	98.51±0.00	**99.24±0.00**	98.82±0.00	96.25±0.00
Promoters	92.45±0.00	75.47±0.00	83.02±0.00	91.51±0.00	**94.34±0.00**	85.85±0.00	77.36±0.00
Soybean	**100.0±0.00**	97.87±0.00	97.87±0.00	**100.0±0.00**	**100.0±0.00**	**100.0±0.00**	97.87±0.00
Spect	**79.09±0.00**	55.62±0.00	75.40±0.00	76.47±0.00	78.07±0.00	78.61±0.00	74.87±0.00
Splice	94.61±0.12	83.69±0.25	94.16±0.15	95.97±0.36	**96.35±0.15**	92.66±0.20	94.45±0.16
Tic-tac-toe	97.11±0.27	98.24±0.19	85.57±0.52	**98.27±0.10**	97.45±0.35	93.58±0.72	97.64±0.16
Voting	95.39±0.60	92.09±0.26	95.06±0.27	**96.14±0.55**	95.82±0.27	95.72±0.50	95.35±0.43
Zoo	**100.0±0.00**	91.09±0.00	98.02±0.00	**100.0±0.00**	**100.0±0.00**	97.03±0.00	99.01±0.00
Average	**91.68±0.21**	85.47±0.14	86.68±0.18	90.36±0.15	**91.26±0.16**	89.90±0.22	85.79±0.19

TABLE III. THE MEAN PREDICTIVE ACCURACY AND THE STANDARD DEVIATION ON SYNTHETIC DATASETS

Dataset	CRN	IBL(k=11)	C4.5	LMT	RF	PART	RIPPER
Extended_Monks-1	**100.0±0.00**	68.13±0.91	96.97±1.32	98.02±0.97	60.65±0.80	89.52±0.41	**100.0±0.00**
Extended_Monks-2	**81.15±0.40**	64.22±1.00	64.85±0.99	64.35±0.94	64.90±0.00	59.41±1.22	64.35±0.82
Extended_Monks-3	**100.0±0.00**	72.52±0.60	**100.0±0.00**	**100.0±0.00**	63.50±0.22	**100.0±0.00**	**100.0±0.00**
RDG10000-200	**92.46±0.25**	66.09±0.26	89.92±0.14	88.58±0.35	75.09±0.15	89.39±0.50	**92.46±0.21**
RDG10000-200-50	**92.49±0.09**	72.03±0.07	90.38±0.13	90.04±0.19	72.26±0.10	89.99±0.12	90.71±0.13
RDG3000-1000	82.64±0.23	62.70±0.31	74.23±0.44	79.83±0.58	64.43±0.03	67.73±0.34	**93.17±0.26**
RDG3000-1000-500	**83.56±0.21**	47.43±0.58	76.17±0.48	80.30±0.42	53.60±0.58	77.07±1.01	83.03±0.46
Average	**90.33±0.17**	64.73±0.53	84.65±0.50	85.87±0.49	64.92±0.27	81.87±0.51	89.10±0.27

All datasets used in our experiments are categorical data and summarized in Table I, among which seventeen datasets are benchmark datasets for machine learning and publicly available from UCI machine learning repository [13]. The others are synthetic data that are used to test the performance of CRN and other classifiers when many irrelevant attributes are added into a dataset. Another purpose is to test whether the quality measure (2) may be biased by attributes with a great number of values. The detailed information about the seventeen UCI datasets can be found in [13], so we only make a further introduction about the synthetic datasets.

The datasets *Extended_Monks*-1, *Extended_Monks*-2, and *Extended_Monks*-3 are obtained by adding ninety-four attributes into the UCI datasets *Monks*-1, *Monks*-2, and *Monks*-3, and the target concept is the same one in *Monks*-1, *Monks*-2, and *Monks*-3, respectively. All added ninety-four attributes have 2 to 31 distinct values with identical probability.

The four datasets *RDG** are generated using the Random Data Generator (abbr. RDG) implemented in the WEKA [14]. The parameters (those are not given in Table I) used to generate the above four datasets are given as follows: all attributes are binary, the maximum number of attribute-value pairs in rules is ten, and the minimum number of attribute-value pairs is one. The difference between *RDG10000-200* (*RDG3000-1000*) and *RDG10000-200-50* (*RDG3000-1000-500*) is that the number of irrelevant attributes is zero for the former and fifty (five hundreds) for the latter.

We use the WEKA [14] as the workbench for classifiers compared with CRN (the version is 3.5.8). Since CRN possesses characteristics of both rule-based classifiers and instance-based classifiers, and a decision tree can be easily converted into a set of rules, we compare the following popular and state-of-the-art classifiers with CRN: the instance-based classifier IBL [1], the decision tree classifiers C4.5 [7], LMT [15], and Random Forests [16], and the rule-based classifiers PART [17] and RIPPER [18]. Considering the space limitation, here we do not give the parameter settings for those compared classifiers except Random Forest. For other classifiers, the parameter settings are default values set in the WEKA.

For Random Forest, the number of attributes to be used in random selection is the first integer less than $\log_2 n + 1$, the number of trees to be generated is set to 250 except dataset *Led-24*, and the maximum depth of the trees is unlimited. Because of the "out-of-memory" error, the number of trees to be generated is set to 150 for dataset *Led-24*.

Table II summarizes the predictive accuracy and the standard deviation for all classifiers on seventeen UCI datasets, and Table III summarizes the predictive accuracy and the standard deviation for all classifiers on seven synthetic datasets. Except for the datasets *Monks*-1, *Monks*-2, *Monks*-3, *Nursery*, and *Spect*, each value in Table II and Table III denotes the mean and the standard deviation over ten runs, where each run consists of a ten-fold ($n > 400$) or n-fold ($n \leq 400$) cross validation. Since a training set and a test set have been provided in datasets *Monks*-1, *Monks*-2, *Monks*-3, *Nursery*, and *Spect*, the cross-validation is not applied to those datasets. The result with bold in each row indicates the highest one.

For seventeen UCI datasets (Table II), we observe that, except for LMT and Random Forests, the predictive accuracy of CRN is comparable to, or better than that of other classifiers on most datasets. Compared with LMT and Random Forests, CRN is a bit weak, but with comparable predictive accuracy in the sense of mean accuracy over seventeen UCI datasets.

For seven synthetic datasets (Table III), we observe that CRN and RIPPER significantly outperform other classifiers. Specifically, the mean accuracy of Random Forests, IBL and PART decreases about 26%, 20%, and 8% compared with the mean accuracy on UCI datasets respectively. If we compare the mean accuracy over UCI datasets with the mean accuracy over synthetic datasets, we also observed that CRN and C4.5 performs more stably than other classifiers. At the same time, Table III demonstrates that CRN is insensitive to irrelevant attributes and the quality measure (2) is not biased by the attributes with a great number of values.

IV. Conclusion

In this paper, a new instance-based learning algorithm for categorical data, named CRN, is proposed based on a rule-like definition of neighbor. CRN is a parameter-free and nonmetric classifier. CRN together with six state-of-the-art classifiers implemented in the WEKA workbench are evaluated on twenty-four datasets. Experimental results indicate that although the underlying idea is simple, the performance of CRN is comparable to or better than that of those state-of-the-art classifiers.

Acknowledgments

We would like to thank the anonymous reviewers for their useful comments to improve our work. This work was supported by China National Science Foundation under Grant No 60973083, State Key Laboratory of Brain and Cognitive Science under Grant No 08B12, Natural Science Foundation of Guangdong Province under Grant No 06300170, and the Fundamental Research Funds for the Central Universities, SCUT, under Grant No 2009ZM0175, 2009ZZ0045, and 2009ZM0189.

References

[1] D. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms", Machine Learning, vol. 6, 1991, pp. 37–66.

[2] T. Cover, and P. E. Hart, "Nearest neighbor pattern classification", IEEE Trans. Information Theory, vol. 13, 1967, pp. 21–27.

[3] S. Cost, and S. Salzberg, "A weighted nearest neighbor algorithm for learning with symbolic features", Machine Learning, vol. 10, 1993, pp. 57–78.

[4] P. Domingos, "Unifying instance-based and rule-based induction", Machine Learning, vol. 24, 1996, pp. 141–168.

[5] J. Fürnkranz, "Separate-and-conquer rule learning", Artificial Intelligence Review, vol .13, 1999, pp. 3–54.

[6] J. R. Quinlan, "Induction of decision trees", Machine Learning, vol. 1, 1986, pp. 81–106.

[7] J. R. Quinlan, C4.5: Programs for Machine Learning. San Mateo: Morgan Kaufman, 1993.

[8] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and Regression Trees. Monterey: Wadsworth International Group, 1984.

[9] D. Haussler, "Quantifying inductive bias: AI learning algorithms and Valiant's learning framework", Artificial Intelligence, vol. 36, 1988, pp. 177–221.

[10] D. S. Johnson, "Approximation algorithms for combinatorial problems", Journal of Computer and System Science, vol. 9, 1974, pp. 256–278.

[11] U. Feige, "A threshold of ln*n* for approximating set cover", Journal of the ACM, vol .45, 1998, pp. 634–652.

[12] R. O., Duda, P. E. Hart, and D. G. Stork, Pattern Classification, 2nd Edition. New York: John Wiley, 2001.

[13] C.Blake, and C. J. Merz, "UCI repository of machine learning databases", CA: University of California, Irvine, 2002, http:// www.ics.uci.edu/~mlearn/mlrepository.html.

[14] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update", SIGKDD Explorations, vol. 11, 2009, pp. 10–17.

[15] N. Landwehr, M. Hall, and E. Frank, "Logistic model trees", Machine Learning, vol. 95, 2005, pp. 161–205.

[16] L. Breiman, "Random forests". Machine Learning, vol. 45, 2001, pp. 5–32.

[17] E. Frank, and I. H. Witten, "Generating accurate rule sets without global optimization", Proc. of the 15th International Conference on Machine Learning (ICML 98), San Francisco: Morgan Kaufmann, 1998, pp. 144–151.

[18] W. W. Cohen, "Fast effective rule induction", Proc. of the 12th International Conference on Machine Learningn (ICML 95), San Francisco: Morgan Kaufmann, 1995, pp. 115–123.