On the Use of Constructs for Rule-Based Classification: A Case Study

Manuel S. Lazo-Cortés[1,2](✉), José Fco. Martínez-Trinidad[1],
and Jesús A. Carrasco-Ochoa[1]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla, Mexico
{mlazo,fmartine,ariel}@inaoep.mx
[2] SEP/SES/TecNM/Instituto Tecnológico de Tlalnepantla,
Tlalnepantla de Baz, Mexico

Abstract. In Rough Set Theory, super-reducts are subsets of attributes that retain the ability of the whole set of attributes to discern objects belonging to different classes; reducts are minimal ones. On the other hand, constructs also allow discerning objects belonging to different classes but, at the same time, they retain similarities between objects belonging to the same. Therefore, constructs are a kind of super-reducts in whose definition inter-class and intra-class information is combined. This type of super-reduct has been little studied. In this paper, we present a case study, about the use of constructs instead of reducts for building decision rules useful for rule-based classification. Our results show the practical utility of constructs for rule based classification.

1 Introduction

Rough Set Theory makes an effort to examine whether a set of descriptive attributes is sufficient to classify objects into the same classes as the original partition. In this effort, super-reducts play an important role. Rough set theory performs analysis and reasoning about data in a data table, in which rows are objects, columns are attributes, and each cell is the value of an attribute on an object [9,10]. A decision table is a special data table such that the set of attributes is the union of a set of condition attributes and a set of decision attributes (most of the time, only one). The notion of super-reduct plays a fundamental role in rough set analysis. Pawlak [10] defined a super-reduct of a decision table as a subset of condition attributes that has the same classification ability as the entire set of condition attributes with respect to the set of decision attributes. Reducts are minimal super-reducts.

On the other hand, constructs [15] take into account more information, because they preserve discriminating relations between objects belonging to different classes and similarity relations between objects belonging to the same class. So, constructs are a kind of super-reducts in whose definition inter-class and intra-class information is combined.

Combining inter-class dissimilarity with intra-class similarity seems interesting because the resulting subsets of attributes (constructs) would ensure not only

J. A. Carrasco-Ochoa et al. (Eds.): MCPR 2019, LNCS 11524, pp. 327–335, 2019.
https://doi.org/10.1007/978-3-030-21077-9_30

the ability to distinguish objects belonging to different classes but also recognizing objects belonging to the same class. This type of super-reduct has been little studied and to the best of our knowledge, this is the first time the usefulness of constructs is studied for building classification rules.

In this paper, we present a case study about the use of constructs for building decision rules useful for a rule-based classifier. The rest of the document is organized as follows. Section 2 provides the formal definitions of reduct and construct. Section 3 presents a case study showing the experimental results that we obtained applying a rule-based classifier when rules are generated through reducts or constructs. A discussion about these results, as well as a comparison against other well known rule-based classifiers, is included in this section. Our conclusions are summarized in Sect. 4.

2 Theoretical Foundations

In this section, we introduce the definitions of reduct and construct under the same notation.

2.1 Reducts

The main data representation considered in this paper is a decision table, which is a special case of an information table [9]. Formally, a decision table is defined as

Definition 1 *(decision table)***.** *A decision table is a pair* $\mathcal{S}_d = (\mathcal{U}, A_t = A_t^* \cup \{d\})$ *where* $\mathcal{U}$ *is a finite non-empty set of objects,* A_t *is a finite non-empty set of attributes.* A_t^* *is a set of conditional attributes and* d *is a decision attribute indicating the decision class for each object in the universe. Each* $a \in A_t$ *corresponds to the function* $I_a : \mathcal{U} \rightarrow V_a$ *called evaluation function, where* V_a *is called the value set of* a*. The decision attribute allows partitioning the universe into blocks (classes) determined by all possible decisions.*

Sometimes we will use D for denoting $\{d\}$, i.e. $(\{d\} = D)$.

A *decision table* can be implemented as a two-dimensional array (matrix), rows are associated to objects, columns to attributes and cells to values of attributes on objects.

When considering decision tables, it is important to distinguish between the so called *consistent* and *inconsistent* ones. A decision table is said to be *consistent*, if each combination of values of descriptive attributes uniquely determines the value of the decision attribute (i.e. objects for which their value of the decision attribute are different have a different description according to the descriptive attributes); and *inconsistent*, otherwise. For the purpose of this paper we only consider consistent decision tables.

It is important to introduce the definition of the indiscernibility relation.

Definition 2 *(indiscernibility relation)*. *Given a subset of conditional attributes $A \subseteq A_t^*$, the indiscernibility relation is defined as $IND(A|D) = \{(u, v) \in \mathcal{U} \times \mathcal{U} : \forall a \in A, [I_a(u) = I_a(v)] \vee [I_d(u) = I_d(v)]\}$*

The indiscernibility relation is an equivalence relation, so it induces a partition over the universe. Being $\mathcal{S}_d$ a consistent decision table, the partition induced by any subset of conditional attributes is finer than (or at maximum equal to) the relation determined by all possible values of the decision attribute d.

We can find several definitions of reduct (see for example, [8]), nevertheless, according to the aim of this paper, we refer to reducts assuming the classical definition of discerning decision reduct [10] as follows.

Definition 3 *(reduct for a decision table)*. *Given a decision table $\mathcal{S}_d$, an attribute set $R \subseteq A_t^*$ is called a reduct, if R satisfies the following two conditions:*

(i) $IND(R|D) = IND(A_t^*|D)$;
(ii) For any $a \in R, IND((R - \{a\})|D) \neq IND(A_t^*|D)$.

All attribute subsets satisfying condition (i) are called super-reducts.

This definition ensures that a reduct has no lower ability to distinguish objects belonging to different classes than the whole set of attributes, being minimal with regard to inclusion, i.e. a reduct does not contain redundant attributes or, equivalently, a reduct does not include other super-reducts. The original idea of reduct is based on inter-class comparisons.

2.2 Constructs

As noted before, reducts are defined from an inter-class object comparison point of view. They ensure preserving the ability to discern between objects belonging to different classes. The novelty of the concept of construct (introduced by Susmaga in 2003 [15]) is the combination of inter-class and intra-class comparisons in such a way that a resulting subset of conditional attributes would ensure not only the ability to distinguish objects belonging to different classes, but also preserves certain similarity between objects belonging to the same class.

Let us now consider the following similarity relation defined between objects belonging to the same class in a decision table $\mathcal{S}_d = (\mathcal{U}, A_t^* \cup \{d\})$.

Definitions 4 and 5 were introduced by Susmaga [15], here we reformulate them for homogeneity in the notation.

Definition 4 *(similarity relation)*. *Given a subset of conditional attributes $A \subseteq A_t^*$, the similarity relation is defined as $SIM(A|D) = \{(u, v) \in \mathcal{U} \times \mathcal{U} : [I_d(u) = I_d(v)]$ and $\exists a \in A\ [I_a(u) = I_a(v)]\}$.*

If a pair of objects belongs to $SIM(A|D)$ then these objects belong to the same class and they are indiscernible on at least one attribute from the set A.

The definition of construct may be stated as follows.

Definition 5 *(construct). Given a decision table $\mathcal{S}_d$, an attribute set $C \subseteq A_t^*$ is called a construct, if C satisfies the following conditions:*

(i) $IND(C|D) = IND(A_t^*|D)$;
(ii) $SIM(C|D) = SIM(A_t^*|D)$;
(iii) *For any* $a \in C, IND((C-\{a\})|D) \neq IND(A_t^*|D)$ *or* $SIM((C-\{a\})|D) \neq SIM(A_t^*|D)$;

So, a construct is a subset of attributes that retains the discernment between any pair of objects belonging to different classes as well as the similarity of objects belonging to the same class. Alike reducts, a construct is minimal, which means that removing any attribute from it would result in making any (or both) of the conditions given by (i) and (ii) invalid.

Example 1. Given the decision table M, where $\mathcal{U} = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$, $A_t^* = \{a_1, a_2, a_3, a_4\}$ and $D = \{d\}$.

$$M = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \end{array} \begin{array}{c} \begin{array}{ccccc} a_1 & a_2 & a_3 & a_4 & d \end{array} \\ \left(\begin{array}{ccccc} \mathit{blue} & 0 & TX & 0 & 1 \\ \mathit{blue} & 1 & NY & 1 & 1 \\ \mathit{white} & 0 & NY & 2 & 1 \\ \mathit{white} & 1 & TX & 0 & 1 \\ \mathit{blue} & 1 & IL & 3 & 2 \\ \mathit{black} & 1 & FL & 4 & 2 \\ \mathit{red} & 1 & AL & 3 & 3 \\ \mathit{red} & 0 & IL & 1 & 3 \end{array}\right) \end{array}$$

From this decision table, we have that $\{a_2, a_3\}$, $\{a_1, a_4\}$, $\{a_1, a_3\}$ and $\{a_3, a_4\}$ are the reducts. Notice that $\{a_2, a_3\}$ does not fulfill the definition of construct, since $0 = I_{a_2}(u_3) \neq I_{a_2}(u_4) = 1$ and $NY = I_{a_3}(u_3) \neq I_{a_3}(u_4) = TX$ being $I_d(u_3) = I_d(u_4) = 1$. In fact, none of the reducts is a construct for this decision table; the only construct for M is $\{a_1, a_2, a_3\}$. $\{a_1, a_2, a_3\}$ is a super-reduct because it contains $\{a_1, a_3\}$ which is a reduct (it also contains $\{a_2, a_3\}$), therefore $\{a_1, a_2, a_3\}$ fulfills condition (i) in Definition 5. For each pair of objects in the same class, we have that $\mathit{blue} = I_{a_1}(u_1) = I_{a_1}(u_2)$, $0 = I_{a_2}(u_1) = I_{a_2}(u_3)$, $TX = I_{a_3}(u_1) = I_{a_3}(u_4)$, $NY = I_{a_3}(u_2) = I_{a_3}(u_3)$, $1 = I_{a_2}(u_2) = I_{a_2}(u_4)$, $\mathit{white} = I_{a_1}(u_3) = I_{a_1}(u_4)$, $1 = I_{a_2}(u_5) = I_{a_2}(u_6)$ and $\mathit{red} = I_{a_1}(u_7) = I_{a_1}(u_8)$. Therefore, $\{a_1, a_2, a_3\}$ fulfills condition (ii) in Definition 5. Finally it is not difficult to verify that $\{a_1, a_2, a_3\}$ is minimal.

3 Case Study

In this section, we will show a case study about the use of constructs instead of reducts for building decision rules useful for a rule-based classifier.

To build the set of decision rules to be used in a rule-based classifier, we used the tools included in the software RSES ver. 2.2.2 [3], which has been widely used in the literature, see for example [2,11,12].

In RSES, once the reducts of a decision table have been computed, each object in the training sample is matched against each reduct. This matching gives as result a rule having in its conditional part, the attributes of the reduct, each one associated with the values of the currently considered object, and in its decision part it has the class for this training object.

At classifying an unseen object through the generated rule set, it may happen that several rules suggest different decision values. In such conflict situations a strategy to reach a final result (decision) is needed. RSES provides a conflict resolution strategy based on voting. In this method, when the antecedent of the rule matches the unseen object, a vote in favor of the decision value of its consequent is cast. Votes are counted and the decision value reaching the majority of the votes is chosen as the class for the object.

This simple method may be extended by assigning weights to rules. In RSES, this method (known as Standard Voting) assigns as weight for a rule the number of training objects matching the antecedent of this rule. Then, each rule votes with its weight and the decision value reaching the highest weight sum is considered as the class for the object.

In the same way that was explained above for reducts, in order to obtain a set of decision rules based on constructs, RSES was used. This was done by loading the set of constructs as if they were reducts, with the format corresponding to this type of file.

For our case study, we used the lymphography dataset, taken from the UCI Machine Learning Repository [1]. We selected this dataset to compare the results with those reported in [7]. We randomly generated two folds in order to perform two-fold cross validation. Characteristics of both the lymphography dataset and the folds used in our experiments can be seen in Table 1.

We used the sets of all reducts and all constructs for creating decision rules. Reducts and rules were computed by using RSES [14]. Constructs were computed by using the typical testor computation algorithm CT-EXT [13], following [6].

Table 1. Characteristics of the lymphography dataset and the two folds used in our experiments

Attributes	Classes	Objects	Objects per class			
			K_1	K_2	K_3	K_4
18	4	148	2	81	61	4
	Fold 1	74	1	40	31	2
	Fold 2	74	1	41	30	2

For each fold, we compute the sets of all reducts and all constructs. Table 2 shows the number of reducts and constructs computed for each fold.

The number of reducts and constructs is large enough to make it difficult to select the best ones. At this point, it is important to emphasize that, from a

Table 2. Number of reducts and constructs for each fold of the lymphography dataset

Fold	Reducts	Constructs
1	530	1431
2	317	970

practical point of view, the simple number of reducts and constructs is an informative indicator of the quality of the regularities found in the data. Although the reducts and constructs are few, the regularities can be strong. On the other hand, when the amount of reducts or constructs become large, the reducts and constructs generated are usually of low quality, since they tend to be a large number of combinations of attributes that satisfy the definitions. Of course, it is still possible that some of these reducts and constructs are really good, but detecting them is difficult due to the search in a large set is time consuming.

In Table 3, it can be seen the minimum, average, and maximum length of reducts and constructs, for both folds.

Table 3. Length of reducts and constructs for the lymphography dataset

Fold	Length	Reducts	Constructs
1	Minimum	5	6
	Average	7.3	8.1
	Maximum	10	11
2	Minimum	5	6
	Average	7	7.8
	Maximum	9	10

We can observe that, as previously reported in [15], this dataset produces more constructs than reducts, and also the constructs tend to contain more attributes than the reducts.

We generate a set of reduct-based rules for the set of all reducts, as well as a set of construct-based rules considering all constructs. Table 4 shows information about the sets of rules. The third column contains the number of rules, the subsequent four columns show the number of rules per class; and the three final columns show the minimum, average and maximum number of objects matching the antecedent of the rules (support of the rules). As it can be seen in these last three columns, for no rule more than seven objects matched the antecedent of the rule. Moreover, from the average (penultimate column), it can be seen that for most rules only one object matched the antecedent of the rule.

We apply, over the two folds, the RSES Standard Voting rule-based classifier and compute the average of the classification accuracy obtained in each fold.

Table 4. Characteristics of the rules generated for the lymphography dataset

Fold	Set	Rules	Rules per class				Support		
			K_1	K_2	K_3	K_4	Min	Mean	Max
1	Reducts	34958	530	18411	14957	1060	1	1.1	6
	Constructs	95797	1431	50585	41194	2862	1	1.1	6
2	Reducts	21138	317	11578	8620	623	1	1.1	5
	Constructs	62759	970	34913	25742	1914	1	1.1	7

Table 5 shows the results obtained in terms of accuracy in average when reduct-based rules were used in the Standard Voting classifier. Additionally, in Table 5, we show the confusion matrix as well as the true positive rate and the accuracy obtained by the Standard Voting rule-based classifier for each class. On average, the Standard Voting rule-based classifier using reduct-based rules obtained an accuracy of 0.73.

Table 5. Confusion matrix for the Standard Voting rule-based classifier for the lymphography dataset using reducts

	K_1	K_2	K_3	K_4	No. of objects	Accuracy
K_1	0	1	1	0	2	0.00
K_2	2	62	14	3	81	0.77
K_3	1	14	46	0	61	0.75
K_4	0	2	1	0	4	0.00
Total					148	**0.73**
True positive rate	0.00	0.78	0.74	0.00		

We repeat the procedure, but now considering constructs instead of reducts. Table 6 shows the results obtained in terms of accuracy in average when construct-based rules were used in the Standard Voting classifier. On average, the Standard Voting rule-based classifier using reduct-based rules obtained an accuracy of 0.78.

As we can see from Tables 5 and 6, when considering rules generated from constructs instead of reducts, the classification accuracy was improved.

Finally, taking into account that we are evaluating the practical utility of using constructs for a rule-based classifier, we wanted to compare the results obtained by rule-based classifiers based on reducts or constructs against those obtained with other well-known rule-based classifiers widely used in the literature. We select RIPPER [4] and SLIPPER [5]. These classifiers were run using the KEEL Software Suite [16].

Table 6. Confusion matrix for the Standard Voting rule-based classifier for the lymphography dataset using constructs

	K_1	K_2	K_3	K_4	No. of objects	Accuracy
K_1	0	1	1	0	2	0.00
K_2	2	69	9	1	81	0.85
K_3	2	13	46	0	61	0.75
K_4	0	1	2	0	4	0.00
Total					148	**0.78**
True positive rate	0.00	0.82	0.73	0.00		

Table 7 shows the results obtained by each compared classifier in ascending order. As we can see construct-based Standard Voting classifier got the best result.

Table 7. Accuracy of four rule-based classifiers for the lymphography dataset

Algorithm	Accuracy
RIPPER	0.69
Reducts based Standard Voting	0.73
SLIPPER	0.76
Constructs based Standard Voting	**0.78**

4 Conclusions

As we have discussed along the paper, reducts and constructs constitute two different contributions to the attribute reduction problem in Rough Set Theory.

The main purpose of the research reported in this paper is the discussion through a case study of the possible advantages that we can obtain when using constructs instead of reducts, for generating classification rules. Our experimental results allow concluding that constructs are an alternative for building rules which can improve the classification accuracy of the rules built from reducts. Even more, the classification results are better than other rule-based classifiers widely used in the literature. These results motivate to delve into the advantages of using either reducts or constructs, specially it may be interesting to study the development of methods to generate rules from a subset of reducts or constructs instead of considering the rules generated by the whole set.

References

1. Bache, K., Lichman, M.: UCI machine learning repository. University of California, School of Information and Computer Science, Irvine, CA (2013). http://archive.ics.uci.edu/ml
2. Barman, T., Rajesh, G., Archana, R.: Rough set based segmentation and classification model for ECG. In: Conference on Advances in Signal Processing (CASP), pp. 18–23. IEEE (2016)
3. Bazan, J.G., Szczuka, M.: The rough set exploration system. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets III. LNCS, vol. 3400, pp. 37–56. Springer, Heidelberg (2005). https://doi.org/10.1007/11427834_2
4. Cohen, W.W.: Fast effective rule induction. In: Proceedings of the Twelfth International Conference on Machine Learning, Lake Tahoe California, USA, pp. 1–10 (1995)
5. Cohen, W.W., Singer, Y.: A simple, fast, and effective rule learner. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence, Orlando Florida, USA, pp. 335–342 (1999)
6. Lazo-Cortés, M.S., Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F., Sanchez-Diaz, G.: Computing constructs by using typical testor algorithms. In: Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F., Sossa-Azuela, J.H., Olvera López, J.A., Famili, F. (eds.) MCPR 2015. LNCS, vol. 9116, pp. 44–53. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19264-2_5
7. Lazo-Cortés, M.S., Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A.: Class-specific reducts vs. classic reducts in a rule-based classifier: a case study. In: Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A., Olvera-López, J.A., Sarkar, S. (eds.) MCPR 2018. LNCS, vol. 10880, pp. 23–30. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-92198-3_3
8. Miao, D.Q., Zhao, Y., Yao, Y.Y., Li, H.X., Xu, F.F.: Reducts in consistent and inconsistent decision tables of the Pawlak rough set model. Inf. Sci. **179**(24), 4140–4150 (2009)
9. Pawlak, Z.: Rough sets. Int. J. Comput. Inf. Sci. **11**, 341–356 (1982)
10. Pawlak, Z.: Rough sets. In: Theoretical Aspects of Reasoning About Data, pp. 315–330. Kluwer Academic Publishers, Dordrecht (1992)
11. Rana, H., Lal, M.: A rough set theory approach for rule generation and validation using RSES. Int. J. Rough Sets Data Anal. **3**(1), 55–70 (2016)
12. Rana, H., Lal, M.: A comparative study based on rough set and classification via clustering approaches to handle incomplete data to predict learning styles. Int. J. Decis. Support Syst. Technol. **9**(2), 1–20 (2017)
13. Sanchez-Diaz, G., Lazo-Cortes, M., Piza-Davila, I.: A fast implementation for the typical testor property identification based on an accumulative binary tuple. Int. J. Comput. Intell. Syst. **5**(6), 1025–1039 (2012)
14. Skowron, A., Bazan, J., Szczuka, M., Wroblewski, J.: Rough set exploration system (version 2.2.2). http://logic.mimuw.edu.pl/~rses/
15. Susmaga, R.: Reducts versus constructs: an experimental evaluation. Electron. Notes Theor. Comput. Sci. **82**(4), 239–250 (2003)
16. Triguero, I., et al.: KEEL 3.0: an open source software for multi-stage analysis in data mining. Int. J. Comput. Intell. Syst. **10**, 1238–1249 (2017)