Hybrid Deletion Policies for Case Base Maintenance

Maria Salamó and Elisabet Golobardes

Enginyeria i Arquitectura La Salle, Universitat Ramon Llull,
Psg. Bonanova 8, 08022 Barcelona, Spain
{mariasal,elisabet}@salleurl.edu

Abstract. Case memory maintenance in a Case-Based Reasoning system is important for two main reasons: (1) to control the case memory size; (2) to reduce irrelevant and redundant instances that may produce noise in the Case-Based Reasoning system. In this paper we present two approaches based on deletion policies to the maintenance of case memories. The foundations of both approaches are the Rough Sets Theory, but each one applies a different policy to delete or maintain cases. The main purpose of these methods is to maintain the competence of the system and reduce, as much as possible, the size of the case memory. Experiments using different domains, most of them from the UCI repository, show that the reduction techniques maintain the competence obtained by the original case memory. The results obtained are compared with those obtained using well-known reduction techniques.

1 Introduction and Motivation

Case-Based Reasoning (CBR) systems solve problems by reusing the solutions to similar problems stored as cases in a case memory [9] (also known as case-base). However, these systems are sensitive to the cases present in the case memory and often its good competence depends on the significance of the cases stored.

The aim of this paper is twofold: (1) to remove noisy cases and (2) to achieve a good generalisation accuracy. This paper presents two hybrid deletion techniques based on Rough Sets Theory. In a previous paper, we presented two reduction techniques based on these measures [10]. This paper continues the initial approaches presented in the previous one, defining a competence model based on Rough sets and presenting new hybrid approaches to improve the weak points. The conclusion of the previous work was that the proposed reduction techniques were complementary, so hybrid methods will achieve a higher reduction and better competence case memories. Thus, in this paper, we present two hybrid approaches: Accuracy-Classification Case Memory (ACCM) and Negative Accuracy-Classification Case Memory (NACCM). Both reduction techniques have been introduced into our Case-Based Classifier System called BASTIAN.

The paper is structured as follows: section 2 introduces related work; next, section 3 explains the foundations of Rough Sets Theory used in our reduction

techniques; section 4 details the proposed reduction techniques based on deletion policies; section 5 describes the testbed of the experiments and the results obtained; and finally, section 6 presents the conclusions and further work.

2 Related Work

Case-Based Reasoning systems solve problems by reusing a corpus of previous solving experience stored as a case memory T of solved cases t. A performance goal for any practical CBR system is the maintenance of a case memory T maximizing coverage and minimizing case memory storage requirements.

Many researchers have addressed the problem of case memory reduction [15, 14] and different approaches have been proposed. The first kind of approaches are *nearest neighbours editing* rules (CNN,SNN,DEL,ENN,RENN). The second kind of approaches are related to Instance Based Learning Algorithms (IBL) [1]. Another approach to instance pruning systems are those that take into account the order in which instances are removed (DROP1 to DROP5)[15].

Another way to approach this problem is to modify the instances themselves, instead of simply deciding which ones to keep. RISE [3] treats each instance as a rule that can be generalised. EACH [11] introduced the *Nested Generalized Exemplars* (NGE) theory, in which hyperrectangles are used to replace one or more instances, thus reducing the original training set.

Finally, researchers have also focused on increasing the overall competence, *the range of target problems that can be successfully solved*, of the case memory through case deletion [12]. Strategies have been developed for controlling case memory growth through methods such as competence-preserving deletion [12] and failure-driven deletion [8], as well as for generating compact case memories through competence-based case addition [17, 13]. Leake and Wilson [5] examine the benefits of using fine-grained performance metrics to directly guide case addition or deletion. This method is specially important for task domains with non-uniform problem distributions. Finally, a case-base maintenance method that avoids building sophisticated structures around a case-base or complex operations is presented by Yang and Wu [16]. Their method partitions cases into clusters where the cases in the same cluster are more similar than cases in other clusters. Clusters can be converted to new smaller case-bases.

3 Rough Sets theory

Zdzislaw Pawlak introduced Rough Sets theory in 1982 [7]. The idea of Rough Sets consists of the approximation of a set by a pair of sets, called the lower and the upper approximation of this set. These approximations are generated by the available data about the elements of the set.

We use Rough Sets theory for extracting the dependencies in knowledge. These dependencies are the basis for computing the relevance of instances into the Case-Based Classifier System. We use two measures of case relevance to decide later which cases have to be deleted from the case memory applying

different policies. First measure is ***Accuracy Rough Sets***, that captures the *degree of completeness* of our knowledge. Second one is ***Class Rough Sets***, that computes the *quality of approximation* of each case. Next sections introduce some concepts and definitions and how to extract these two measures.

3.1 Introduction to the Rough Sets Theory

We have a **Universe** (U) (finite not null set of cases that describes our problem, i.e. the case memory). We compute from our universe the **concepts** (cases) that form partitions. The union of all the *concepts* make the entire Universe. Using *all the concepts* we can describe all the **equivalence relations** (R) over the universe U. Let an equivalence relation be a *set of features* that describe a specific concept. U/R is the family of all **equivalence classes** of R. The universe and the relations form the **knowledge base** (K), defined as $K =< U, \hat{R} >$. Where $\hat{R}$ is the family of equivalence relations over U. Every relation over the universe is an elementary concept in K. All the concepts are formed by a set of equivalence relations that describe them. Thus, we search for the minimal set of R that defines the same concept as the initial set.

Definition 1 (Indiscernibility Relations)
$IND(\hat{P}) = \bigcap \hat{R}$ where $\hat{P} \subseteq \hat{R}$. The indiscernibility relation is an equivalence relation over U. Hence, it partitions the concepts (cases) into equivalence classes. These sets of classes are sets of instances indiscernible with respect to the features in P. Such a partition is denoted as $U/IND(P)$. In supervised machine learning, the sets of cases indiscernible with respect to the class attribute contain the cases of each class.

Approximations of Set. Given a condition set that contains all cases present in the case memory and a decision set that presents all the classes that the condition set has to classify. We are searching for a subset of the condition set able to classify the same as the initial set, so it approximates the same decision set. The following definitions explain this idea.

Let $K =< U, \hat{R} >$ be a knowledge base. For any subset of cases $X \subseteq U$ and an equivalence relation $R \in \hat{R}$, $R \subseteq IND(K)$ we associate two subsets called: Lower $\underline{R}X$; and Upper $\overline{R}X$ approximations. If $\underline{R}X$=$\overline{R}X$ then X is an *exact set* (definable using subset R), otherwise X is a ***rough set*** with respect to R.

Definition 2 (Lower approximation)
The lower approximation, defined as: $\underline{R}X = \bigcup\{Y \in U/R : Y \subseteq X\}$ is the set of all elements of U which can *certainly* be classified as elements of X in knowledge R.

Definition 3 (Upper approximation)
The upper approximation, $\overline{R}X = \bigcup\{Y \in U/R : X \bigcap Y \neq \emptyset\}$ is the set of elements of U which can *possibly* be classified as elements of X, employing knowledge R.

Reduct and Core of knowledge This part is related to the concept of reduction of the feature search space that defines the initial knowledge base. Next, this reduced space is used to extract the relevance of each case. Intuitively, a **reduct** of knowledge is its essential part which suffices to define all concepts occurring in the knowledge, whereas the **core** is the most important part.

Let $\hat{R}$ be a family of equivalence relations and $R \in \hat{R}$. We will say that:

- R is *indispensable* if $IND(\hat{R}) \neq IND(\hat{R} - \{R\})$; otherwise it is *dispensable*. $IND(\hat{R} - \{R\})$ is the family of equivalence $\hat{R}$ extracting R.
- The family $\hat{R}$ is *independent* if each $R \in \hat{R}$ is *indispensable* in R; otherwise it is *dependent*.

Definition 4 (Reduct)
$\hat{Q} \in \hat{R}$ is a reduct of $\hat{R}$ if : $\hat{Q}$ is *independent* and $IND(\hat{Q}) = IND(\hat{R})$. Obviously, $\hat{R}$ may have many reducts. Using $\hat{Q}$ it is possible to approximate the same as using $\hat{R}$. Each reduct has the property that a feature can not be removed from it without changing the indiscernibility relation.

Definition 5 (Core)
The set of all indispensable relations in $\hat{R}$ will be called the *core* of $\hat{R}$, and will be denoted as: $CORE(\hat{R}) = \bigcap RED(\hat{R})$. Where $RED(\hat{R})$ is the family of all reducts of $\hat{R}$. It is the most characteristic part of knowledge and can not be eliminated.

3.2 Measures of relevance based on Rough Sets

Accuracy Rough Sets and *Class Rough Sets* measures use the information of reducts and core to compute the relevance of each case.

Accuracy Rough Sets This measure computes the *Accuracy* coefficient (**AccurCoef**) of each case t in the knowledge base (case memory T) as:

$$For\ each\ instance\ t\ \in\ T\ it\ computes : AccurCoef(t) = \frac{card\ (\ \underline{P}(t))}{card\ (\ \overline{P}\ (t))} \quad (1)$$

Where $AccurCoef(t)$ is the relevance of the instance t; T is the training set; $card$ is the cardinality of one set; P is the set that contains the *reducts* and *core* obtained from the original data; and finally $\underline{P}(t)$ and $\overline{P}(t)$ are the presence of t in the lower and upper approximations, respectively.

The accuracy measure expresses the degree of completeness of our knowledge about the set P. The accuracy coefficient explains if an instance is on an internal region or on a border line region, so $AccurCoef(t)$ is a binary value. When the value is 0 it means an internal case, and a value of 1 means an outlier case. Inexactness of a set of cases is due to the existence of a borderline region. The greater a borderline region of a set, the lower the accuracy of the set. The accuracy expresses the percentage of possible correct decisions when classifying cases employing knowledge P.

Class Rough Sets In this measure we use the *quality of classification* coefficient (**ClassCoef**). It is computed as:

$$For\ each\ instance\ t\ \in\ T\ it\ computes : ClassCoef(t) = \frac{card\ (\ \underline{P}(t))}{card\ (T)} \quad (2)$$

Where $ClassCoef(t)$ is the relevance of the instance t; T is the training set; $card$ is the cardinality of a set; P is a set that contains the reducts and core; and finally $\underline{P}(t)$ is the presence of t in the lower approximation.

The $ClassCoef$ coefficient expresses the percentage of cases which can be correctly classified employing the knowledge t. This coefficient has a range of values between 0 to 1, where 0 and 1 means that the instance classifies incorrectly and correctly, respectively, the range of cases that belong to its class. The higher the quality, the nearer to the outlier region.

4 Reduction Techniques

In this section, we present two hybrid reduction techniques based on the Rough Sets measures described in section 3.2. The difference between them are the usage of the coverage to select the set of cases which are deleted from the original case memory, in order to achieve a better competence and compact case memory.

4.1 Categorisation model of case memory

The aim of these reduction techniques is to take advantage of the benefits of each coverage measure ($AccurCoef$ and $ClassCoef$). In order to make easier understanding the algorithms and the environment of application, we introduce different concepts and definitions. We use these techniques on classification tasks. For this reason, we modify some definitions.

The distribution of the case memory is done using a new categorisation in terms of their *coverage* and *reachability*. The *coverage* and *reachability* concepts are modified with regard to [12]. However, we maintain as much as possible the essence of the original ones, but it is modified to our coverage measures (explained in 3.2) and to our problem task.

Definition 6 (Coverage)
Let $T = \{t_1, t_2, ..., t_n\}$ be a training set of instances, $\forall\ t_i \in T$:
$Coverage(t_i) = AccurCoef(t_i) \bigoplus ClassCoef(t_i)$

Definition 7 (Reachability)
Let $T = \{t_1, t_2, ..., t_n\}$ be a training set of instances, $\forall\ t_i \in T$:

$$Reachability(t_i) = \begin{cases} class\ (t_i) & if\ it\ is\ a\ classification\ task \\ adaptable(t_i, T)\ if\ it\ is\ not\ a\ classification\ task \end{cases} \quad (3)$$

Where $class(t_i)$ is the class that classifies case t_i.

4.2 Accuracy-Classification Case Memory (ACCM)

Once we have computed the AccurCoef and ClassCoef, we apply for the original case memory the algorithm 1 to select the cases that have to be deleted from the case memory. The cases not selected are maintained in the case memory.

The main idea of this reduction technique is to take benefit from the advantages of both measures separately. First, it maintains all the cases that are outliers, so cases with an AccurCoef = 1.0 value are not removed. This assumption is done because if a case is isolated, there is not any other case that can solve it. Second, the cases selected are those that are nearest to the outliers and other cases nearby can be used to solve it because their coverage is higher.

Algorithm 1 ACCM

1. SelectCasesACCM (CaseMemory T)
2. confidenceLevel = 1.0 and freeLevel = ConstantTuned *(set at 0.01)*
3. select all instances $t \in T$ as $SelectCase(t)$ if accomplish: $coverage(t) \geq$ confidenceLevel
4. **while** not $\exists$ at least a t in $SelectCase$ for each class c that $reachability(t) = c$
5. confidenceLevel = confidenceLevel - freeLevel
6. select all instances $t \in T$ as $SelectCase(t)$ if accomplish: $coverage(t) \geq$ confidenceLevel
7. **end while**
8. delete from CaseMemory the set of cases selected as $SelectCase$
9. **return** CaseMemory T

4.3 Negative Accuracy-Classification Case Memory - (NACCM)

This reduction technique is based on the previous one. The motivation for this technique is to select a wider range of cases than ACCM technique. The main process in ACCM is to select all the cases that are near to the outliers and maintain those cases that are completely internal and does not have any case whose competence are contained. In NACCM the process is to select cases to be maintained in the case memory until all the classes contain almost one case. The NACCM algorithm is divided in two steps: **Step 1** convert coverage measure of each case to its negation measure in order to let us to modify the selection process from internal to outliers points. **Step 2** use the algorithm 2 that describes the SelectCasesNACCM process.

Algorithm 2 NACCM

1. SelectCasesNACCM (CaseMemory T)
2. confidenceLevel = 1.0 and freeLevel = ConstantTuned *(set at 0.01)*
3. select all instances $t \in T$ as $SelectCase(t)$ if t accomplish: $coverage(t) \geq$ confidenceLevel
4. **while** not $\exists$ at least a t in $SelectCase$ for each class c that $reachability(t) = c$
5. confidenceLevel = confidenceLevel - freeLevel
6. select all instances $t \in T$ as $SelectCase(t)$ if t accomplish: $coverage(t) \geq$ confidenceLevel
7. **end while**
8. Maintain in CaseMemory the set of cases selected as $SelectCase$, those cases not selected are deleted from CaseMemory
9. **return** CaseMemory T

Thus, the selection of cases starts from internal cases to outliers ones. The aim is to maintain the minimal set of cases in the case memory. The behaviour of this reduction technique will be very similar to ACCM, but NACCM allows less cases to be maintained in the case memory.

5 Experimental study

This section is structured as follows: first, we describe the testbed used in the experimental study; then we discuss the results obtained from the reduction techniques based on Rough Sets. We analyse the results of our reduction techniques compared to CBR system working with the original case memory. And finally, we also compare the results with some related learning systems.

5.1 Testbed

In order to evaluate the performance rate, we use ten datasets. Datasets can be grouped in two ways: *public* and *private* (details in table 1). **Public datasets** are obtained from the UCI repository [6]. They are: *Breast Cancer Wisconsin (Breast-w), Glass, Ionosphere, Iris, Sonar and Vehicle*. **Private datasets** [4] come from our own repository. They deal with *diagnosis* of breast cancer and *synthetic* datasets. Diagnosis datasets are *Biopsy* and *Mammogram*. On the other hand, *synthetic* datasets are used to tune up the learning algorithms, because we knew their solutions in advance. *MX11* is the eleven input multiplexer and TAO-*grid* is obtained from sampling the TAO figure using a grid. These datasets were chosen in order to provide a wide variety of application areas, sizes, combinations of feature types, and difficulty as measured by the accuracy achieved on them by current algorithms. The choice was also made with the goal of having enough data points to extract conclusions.

Table 1. Datasets and their characteristics used in the empirical study.

	Dataset	Reference	Samples	Numeric feat.	Symbolic feat.	Classes	Inconsistent
1	*Biopsy*	*BI*	1027	24	-	2	Yes
2	*Breast-w*	*BC*	699	9	-	2	Yes
3	*Glass*	*GL*	214	9	-	6	No
4	*Ionosphere*	*IO*	351	34	-	2	No
5	*Iris*	*IR*	150	4	-	3	No
6	*Mammogram*	*MA*	216	23	-	2	Yes
7	*MX11*	*MX*	2048	-	11	2	No
8	*Sonar*	*SO*	208	60	-	2	No
9	*TAO-Grid*	*TG*	1888	2	-	2	No
10	*Vehicle*	*VE*	846	18	-	4	No

The study described in this paper was carried out in the context of BASTIAN, a *case*-**BA***sed* **S***ys***T***em* **I***n cl***A***ssificatio***N**. All techniques were run using the same set of parameters for all datasets: a 1-Nearest Neighbour Algorithm that uses a list of cases to represent the case memory. Each case contains the set of attributes, the class, the AccurCoef and ClassCoef coefficients. Our goal in this paper is to reduce the case memory. For this reason, we have not focus on the representation used by the system. The retain phase does not store any new case in the case memory. Thus, the learning process is limited to the reduced training set. Finally, weighting methods are not used in this paper in order to test the reliability of our reduction techniques. Further work will consists of testing the influence of these methods in conjunction with weighting methods.

The percentage of correct classifications has been *averaged* over *stratified ten-fold cross-validation* runs, with their corresponding standard deviations. To study the performance we use paired *t-test* on these runs.

5.2 Experimental analysis of reduction techniques

The aim of our reduction techniques is to reduce the case memory while maintaining the competence of the system. This priority guide our deletion policies. That fact is detected in the results. Table 2 shows the results for the IBL's algorithms and the Rough Sets reduction techniques. For example, *Vehicle* dataset

obtains a good competence as well as reduces the case memory, in both reduction techniques. The results related to ACCM show a competence maintenance and improvement in some datasets, but the case memory size has not been reduced too much. That results denote that ACCM is able to remove noisy and redundant cases from the case memory, enabling to improve the competence. NACCM technique shows, as expected in its description due to a more restrictive behaviour, a higher reduction of the case memory. However, the reduction in NACCM is not very large. The behaviour is similar to ACCM. It is due to share both reductions techniques the same foundations. The NACCM obtains higher reduction while producing a competence loss, although it is not a significant loss.

Table 2. Mean percentage of correct classifications (%PA) and mean storage size (%CM). Two-sided paired t-test (p = 0.1) is performed, where a ∘ and • stand for a significant improvement or degradation of the reduction techniques related to the CBR. Bold font indicates the best prediction accuracy.

Ref.	**CBR**		**ACCM**		**NACCM**		**IB2**		**IB3**		**IB4**	
	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM
BI	83.15	100.0	83.65	88.01	**83.66**	99.3	75.77•	26.65	78.51•	13.62	76.46•	12.82
BC	**96.28**	100.0	95.71	77.36	95.72	59.52	91.86•	8.18	94.98	2.86	94.86	2.65
GL	**72.42**	100.0	69.83	74.95	64.48	33.91	62.53•	42.99	65.56•	44.34	66.40•	39.40
IO	**90.59**	100.0	**90.59**	83.77	90.30	56.80	86.61•	15.82	90.62	13.89	90.35	15.44
IR	96.0	100.0	**96.66**	89.03	93.33	42.88	93.98	9.85	91.33•	11.26	**96.66**	12.00
MA	64.81	100.0	**66.34**	89.19	60.18	44.80	66.19	42.28	60.16	14.30	60.03	21.55
MX	78.61	100.0	78.61	99.90	78.61	99.90	**87.07**∘	18.99	81.59	15.76	81.34	15.84
SO	84.61	100.0	86.45	71.71	**86.90**	78.24	80.72	27.30	62.11•	22.70	63.06•	22.92
TG	95.76	100.0	**96.13**∘	97.59	90.25	1.54	94.87•	7.38	95.04•	5.63	93.96•	5.79
VE	67.37	100.0	**69.10**∘	72.35	**69.10**∘	72.35	65.46	40.01	63.21•	33.36	63.68•	31.66

In summary, the results obtained using ACCM and NACCM maintain or even improve in a significance level the competence while reducing the case memory.

Comparing rough sets reduction techniques to IBL, ACCM and NACCM obtain on average a higher generalisation on competence than IBL, as it can be seen in table 2. The performance of IBL algorithms declines, in almost all datasets (e.g. *Breast-w, Biopsy*), when case memory is reduced. CBR obtains on average higher prediction competence than IB2, IB3 and IB4. On the other hand, the mean storage size obtained is higher in our reduction techniques than those obtained using IBL schemes.

To finish the empirical study, we also run additional well-known reduction schemes on the previous data sets. Table 3 compare ACCM to CNN, SNN, DEL, ENN, RENN. Table 4 compare ACCM to DROP1, DROP2, DROP3, DROP4 and DROP5 (a complete explanation of them can be found in [15]). We use the same datasets described above but with different ten-fold cross validation sets.

Table 3 shows that the results of ACCM are on average better than those obtained by the reduction techniques studied. RENN improves the results of ACCM in some data sets (e.g. *Breast-w*) but its reduction on the case memory is lower than ACCM.

In table 4 the results obtained using ACCM and DROP algorithms show that ACCM has better competence for some data sets (e.g. *Biopsy, Breast-w, Ionosphere, Sonar*), although its results are also worse in others (e.g. *Mx11*). The behaviour of these reduction techniques are similar to the previously studied. ACCM obtains a balance behaviour between competence and size. There

are some reduction techniques that obtain best competence for some data sets reducing less the case memory size.

Table 3. Mean percentage of correct classifications (%PA) and mean storage size (%CM). Two-sided paired t-test (p = 0.1) is performed, where a ○ and ● stand for a significant improvement or degradation of our ACCM related to the system compared. Bold font indicates the best prediction accuracy.

Ref.	ACCM		CNN		SNN		DEL		ENN		RENN	
	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM
BI	**83.65**	88.01	79.57●	17.82	78.41●	14.51	82.79●	0.35	77.82●	16.52	81.03●	84.51
BC	95.71	77.36	95.57	5.87	95.42	3.72	96.57○	0.32	95.28	3.61	**97.00**○	96.34
GL	**69.83**	74.95	67.64	24.97	67.73	20.51	64.87●	4.47	68.23	19.32	68.66	72.90
IO	**90.59**	83.77	88.89●	9.94	85.75●	7.00	80.34●	1.01	88.31●	7.79	85.18●	86.39
IR	**96.66**	89.03	96.00	14.00	94.00●	9.93	96.00	2.52	91.33●	8.59	96.00	94.44
MA	**66.34**	89.19	61.04	25.06	63.42●	18.05	62.53●	1.03	63.85●	21.66	65.32	66.92
MX	78.61	99.90	89.01○	37.17	89.01○	37.15	68.99●	0.55	85.05○	32.54	**99.80**○	99.89
SO	**86.45**	71.71	83.26	23.45	80.38	20.52	77.45●	1.12	85.62	19.34	82.74	86.49
TG	96.13	97.59	94.39●	7.15	94.76●	6.38	87.66●	0.26	**96.77**	3.75	95.18	96.51
VE	69.10	72.35	**69.74**	23.30	69.27	19.90	62.29●	2.55	66.91	20.70	68.67	74.56

Table 4. Mean percentage of correct classifications (%PA) and mean storage size (%CM). Two-sided paired t-test (p = 0.1) is performed, where a ○ and ● stand for a significant improvement or degradation of our ACCM approach related to the system compared. Bold font indicates best prediction accuracy.

Ref.	ACCM		DROP1		DROP2		DROP3		DROP4		DROP5	
	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM	%PA	%CM
BI	**83.65**	88.01	76.36●	26.84	76.95●	29.38	77.34●	15.16	76.16●	28.11	76.17●	27.03
BC	95.71	77.36	93.28	8.79	92.56●	8.35	**96.28**	2.70	95.00	4.37	93.28	8.79
GL	**69.83**	74.95	66.39	40.86	69.57	42.94	67.27	33.28	69.18	43.30	65.02●	40.65
IO	**90.59**	83.77	81.20●	23.04	87.73●	19.21	88.89●	14.24	88.02●	15.83	81.20●	23.04
IR	**96.66**	89.03	91.33	12.44	90.00●	14.07	92.66●	12.07	88.67●	7.93	91.33●	12.44
MA	**66.34**	89.19	61.60	42.69	58.33●	51.34	58.51●	12.60	58.29●	50.77	61.60●	42.64
MX	78.61	99.90	87.94○	19.02	**100.00**○	98.37	82.37○	17.10	86.52 ○	25.47	86.52○	18.89
SO	86.45	71.71	84.64	25.05	**87.07**	28.26	76.57●	16.93	84.64●	26.82	84.64●	25.11
TG	**96.13**	97.59	94.76●	8.03	95.23●	8.95	94.49●	6.76	89.41●	2.18	94.76●	8.03
VE	**69.10**	72.35	64.66●	38.69	67.16	43.21	66.21	29.42	68.21	43.85	64.66●	38.69

All the experiments (tables 2, 3 and 4) point to some interesting observations. First, it is worth noting that the individual ACCM and NACCM work well in all data sets, obtaining better results on ACCM because its deletion policy is more conservative. Second, the mean storage obtained using ACCM and NACCM is reduced while maintaining the competence on the CBR system. Finally, the results in all tables suggest that all the reduction techniques work well in some, but not all, domains. This has been termed the *selective superiority problem* [2]. Consequently, future work consists of improving the selection of cases in order to be eliminated or maintained in the case memory while maintaining, as well as ACCM and NACCM techniques, the CBR competence.

6 Conclusions and Further Work

This paper presents two reduction techniques whose foundations are the Rough Sets Theory. The aim of this paper is twofold: (1) to avoid noisy and redundant instances and to obtain compact case memories; and (2) to maintain or improve the competence of the CBR system. Empirical study shows that these reduction techniques produce compact-competent case memories. Although the case memory reduction is not large, the competence of the CBR system is improved or

maintained on average. Thus, the generalisation accuracy on classification tasks is guaranteed.

We conclude that the deletion policies could be improved in some points that will be focused on our further work. First, we can modify the competence model presented in this paper to assure a higher reduction on the case memory. Second, it is necessary to study the influence of the learning process. Finally, we want to analyse the influence of the weighting methods in these reduction techniques.

References

1. D. Aha and D. Kibler. Instance-based learning algorithms. *Machine Learning, Vol. 6*, pages 37–66, 1991.
2. C.E. Brodley. Addressing the selective superiority problem: Automatic algorithm/model class selection. In *Proc. of the 10th Int. Conf. on Machine Learning*, pages 17–24, 1993.
3. P. Domingos. Context-sensitive feature selection for lazy learners. In *AI Review*, volume 11, pages 227–253, 1997.
4. E. Golobardes, X. Llorà, M. Salamó, and J. Martí. Computer Aided Diagnosis with Case-Based Reasoning and Genetic Algorithms. *Knowledge-Based Systems*, (15):45–52, 2002.
5. D. Leake and D. Wilson. Remembering Why to Remember: Performance-Guided Case-Base Maintenance. In *Proc. of the 5th European Workshop on Case-Based Reasoning*, pages 161–172, 2000.
6. C. J. Merz and P. M. Murphy. UCI Repository for Machine Learning Databases [http://www.ics.uci.edu/∼mlearn/MLRepository.html].
7. Z. Pawlak. Rough Sets. In *Int. Journal of Information and Computer Science*, volume 11, 1982.
8. L. Portinale, P. Torasso, and P. Tavano. Speed-up, quality and competence in multi-modal reasoning. In *Proc. of the Third Int. Conf. on Case-Based Reasoning*, pages 303–317, 1999.
9. C.K. Riesbeck and R.C. Schank. *Inside Case-Based Reasoning*. Lawrence Erlbaum Associates, Hillsdale, NJ, US, 1989.
10. M. Salamó and E. Golobardes. Rough sets reduction techniques for case-based reasoning. In *Proc. 4th. Int. Conf. on Case-Based Reasoning*, pages 467–482, 2001.
11. S. Salzberg. A nearest hyperrectangle learning method. *Machine Learning*, 6:277–309, 1991.
12. B. Smyth and M. Keane. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In *Proc. of the Thirteen Int. Joint Conf. on Artificial Intelligence*, pages 377–382, 1995.
13. B. Smyth and E. McKenna. Competence Models and the maintenance problem. *Computational Intelligence*, 17(2):235–249, 2001.
14. D.C. Wilson and D.B. Leake. Maintaining Case-Based Reasoners:Dimensions and Directions. *Computational Intelligence*, 17(2):196–213, 2001.
15. D.R. Wilson and T.R. Martinez. Reduction techniques for Instance-Based Learning Algorithms. *Machine Learning, 38*, pages 257–286, 2000.
16. Q. Yang and J. Wu. Keep it Simple: A Case-Base Maintenance Policy Based on Clustering and Information Theory. In *Proc. of the Canadian AI Conf.*, pages 102–114, 2000.
17. J. Zhu and Q. Yang. Remembering to add: Competence-preserving case-addition policies for case base maintenance. In *Proc. of the Fifteenth Int. Joint Conf. on Artificial Intelligence*, volume 1, pages 234–239, 1999.