Collaborative Case Retention Strategies for CBR Agents

Santiago Ontañón and Enric Plaza

IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia (Spain).
{santi,enric}@iiia.csic.es, http://www.iiia.csic.es

Abstract. Empirical experiments have shown that storing every case does not automatically improve the accuracy of a CBR system. Therefore, several retain policies have been proposed in order to select which cases to retain. However, all the research done in case retention strategies is done in centralized CBR systems. We focus on multiagent CBR systems, where each agent has a local case base, and where each agent can interact with other agents in the system to solve problems in a collaborative way. We propose several case retention strategies that directly deal with the issue of being in a multiagent CBR system. Those case retention strategies combine ideas from the CBR case retain strategies and from the *active learning* techniques. Empirical results show that strategies that use collaboration with other agents outperform those strategies where the agents work in isolation. We present experiments in two different scenarios, the first one allowing multiple copies of one case and the second one only allowing one copy of each case. Although it may seem counterintuitive, we show and explain why not allowing multiple copies of each case achieves better results.

1 Introduction

Maintaining compact and competent case bases has become a main topic of Case Based Reasoning research. The main goal is to obtain a compact case base (with a reduced number of cases) but without losing problem solving accuracy. Moreover, empirical experiments have shown that storing every case does not automatically improve the accuracy of a CBR system [13]. The last process in the CBR cycle (retrieve, reuse, repair and retain) [1] is in charge of deciding which new cases must be retained. When a case is decided to be retained, it is incorporated into the case base and will be accessible for solving new problems in the future.

Deciding which cases to retain (or to select which cases to learn from) is a concern not only in CBR. A main issue on machine learning is to select which are the examples of the target problem to learn from. Each time a learning system receives a new example, it has two options: use the example to learn (retain) or discard it. When a learner retains every example it observes, we are

talking of *passive learning*. But when the learner has some strategy to select which are the examples that it is going to learn from, we are talking of *active learning* [4]. The basic idea in active learning is that the learner receives a set of unlabeled examples and decides which of them are interesting to learn from; then the teacher labels the examples that the learner has found interesting and they are used for learning. The main goal of active learning is to minimize the number of examples needed to learn any task without appreciably degrading the performance.

Therefore, we have two different approaches to the problem of selecting which are the most interesting examples (cases) to learn from: the CBR approach and the active learning approach. The basic idea of both is to perform an active selection process through the instance space with the goal of selecting the best examples (cases) to learn from. However, there are also fundamental differences between them. Specifically, active learning strategies try to minimize the number of questions to the teacher, i.e. active learning strategies try to select which are the interesting examples *before* knowing their solution, to avoid the cost associated with labeling them (asking for the right solution from a teacher). CBR case retention strategies do not try to minimize the cost of asking for the solution of the cases, but assumes that this solution is known, since the retain process is performed after the revise process in the CBR cycle. However, we will show that both approaches can be seen under a common framework.

This work extends our previous work on *Ensemble CBR* [11]. Ensemble CBR focuses on *Multiagent CBR Systems* ($\mathcal{M}$AC) where the agents are able to solve problems individually using CBR methods and where only local case bases are accesible to each individual agent. Problems to be solved by an agent can be sent by an external user or by another agent. The main issue is to find good collaboration strategies among selfinterested CBR agents that can help improving classification accuracy without compromising case base privacy. In this paper we focus on case retention strategies for Ensemble CBR.

When dealing with a multiagent system both CBR case retention and active learning must be reconsidered. If individual CBR agents apply case retention as if they were in isolation they can be losing relevant information. In a multiagent scenario several learning opportunities arise from the collaboration with other agents. Imagine the following situation: an agent A_i has the opportunity to learn a new example P, but decides that P is not interesting to him. But there is another agent A_j in the system that could obtain a great benefit from learning example P. Both agents would benefit from the fact that the agent A_i does not discard the case but instead gives it or sells it to A_j. Moreover, when an agent retains a new example, the agent can poll the other agents to see if there is anyone else also interested in the new example.

The structure of the paper is as follows. We first present in section 2 a more detailed description of the $\mathcal{M}$AC systems. Then, in section 3 we will present a common framework encompassing both active learning and CBR retention strategies. Within this framework, we will present several strategies exploiting the fact that the agents are inside a multiagent system. Finally, section 4 shows

an empirical evaluation of the strategies presented in this paper. The paper closes with related work and conclusions section.

2 Multiagent CBR Systems

Formally, a $\mathcal{M}\mathsf{AC}$ system $\mathcal{M} = \{(A_i, C_i)\}_{i=1...n}$ is composed on n agents, where each agent A_i has a case base C_i. In this framework we restrict ourselves to analytical tasks, i.e. tasks (like classification) where the solution is achieved by selecting from an enumerated set of solutions $K = \{S_1 \dots S_K\}$. A case base $C_i = \{(P_j, S_k)\}_{j=1...N}$ is a collection of problem/solution pairs. Each agent A_i is autonomous and has learning capabilities, i.e. each agent is able to collect autonomously new cases that can be incorporated to its local case base.

Moreover, since we focus on analytical tasks, there is no obvious decomposition of the problem in subtasks. However, collaboration is still interesting because an agent A_i can send a complete problem P to another agent A_j asking for help to solve it. After A_j answers A_i with its own solution for P, A_i can do anything with this solution. A simple way for A_i to use this solution is to compare it with the solution found by itself, if both solutions agree A_i can increase the degree of confidence on the solution found, and if both solutions disagree, maybe it's interesting to send the problem to some other agent to have a third opinion [10].

When an agent A_i asks another agent A_j help to solve a problem the interaction protocol is as follows. First, A_i sends a problem description P to A_j. Second, after A_j has tried to solve P using its case base C_j, it sends back a message that is either `:sorry` (if it cannot solve P) or a solution endorsement record (SER). A SER has the form $\langle\{(S_k, E_k^j)\}, P, A_j\rangle$, where the collection of *endorsing pairs* (S_k, E_k^j) mean that the agent A_j has found E_k^j cases in case base C_j endorsing solution S_k—i.e. there are a number E_k^j of cases that are relevant (similar) for endorsing S_k as a solution for P. Each agent A_j is free to send one or more endorsing pairs in a SER record.

In our framework, agents use a voting mechanism in order to aggregate the information contained in various SERs coming from other agents. This voting scheme is explained in the next section.

2.1 Voting Scheme

The principle behind the voting scheme is that the agents vote for solution classes depending on the number of cases they found endorsing those classes. However, we want to prevent an agent having an unbounded number of votes. Thus, we will define a normalization function so that each agent has one vote that can be for a unique solution class or fractionally assigned to a number of classes depending on the number of endorsing cases.

Formally, let $\mathcal{A}^t$ the set of agents that have submitted their SERs to the agent A_i for problem P. We will consider that $A_i \in \mathcal{A}^t$ and the result of A_i

trying to solve P is also reified as a SER. The vote of an agent $A_j \in \mathcal{A}^t$ for class S_k is

$$Vote(S_k, A_j) = \frac{E_k^j}{c + \sum_{r=1...K} E_r^j}$$

where c is a constant that on our experiments is set to 1. It is easy to see that an agent can cast a fractional vote that is always less than 1. Aggregating the votes from different agents for a class S_k we have ballot $Ballot^t(S_k, \mathcal{A}^t) = \sum_{A_j \in \mathcal{A}^t} Vote(S_k, A_j)$ and therefore the winning solution class is the class with more votes in total.

This voting scheme can be seen as a variation of *Approval Voting* [3]. In *Approval Voting* each agent vote for all the candidates they consider as possible solutions without giving any weight to its votes. In our scheme, *Approval Voting* can be implemented making $Vote(S_k, A_j) = 1$ if $E_k^j \neq 0$ and 0 otherwise. There are two differences between the standard *Approval Voting* and our voting scheme. The first one is that in our voting scheme agents can give a weight to each one of its votes. The second difference is that the sum of the votes of an agent is bounded by 1. Thus we call it *Bounded-Weighted Approval Voting* (BWAV).

The next section presents the *Committee* collaboration strategy, that uses this voting scheme.

2.2 Committee Collaboration Strategy

In this collaboration strategy the agent members of a $\mathcal{M}$AC system $\mathcal{M}$ are viewed as a committee. An agent A_i that has to solve a problem P sends it to all the other agents in $\mathcal{M}$. Each agent A_j that has received P sends a solution endorsement record $\langle \{(S_k, E_k^j)\}, P, A_j \rangle$ to A_i. The initiating agent A_i uses the voting scheme above upon all SERs, i.e. its own SER and the SERs of all the other agents in the multiagent system. The problem's solution is the class with maximum number of votes.

Since all the agents in a $\mathcal{M}$AC system are autonomous CBR agents, they will not have the same problem solving experience (in general, the cases in their case bases will not be the same). This makes it likely that the errors that each agent make in the solution of problems will not be very correlated, i.e. each agent will not err in the same problems. It is well known in machine learning that the combination of the predictions made by several classifiers with uncorrelated errors improves over the individual accuracies of those classifiers [5] ("ensemble effect"). Thus, using the committee collaboration policy an agent can increase its problem solving accuracy because it matches the preconditions of the "ensemble effect".

3 Strategies for Case Retention

In the following sections we will present several strategies that an agent can use to decide which cases to retain. These strategies are used when an agent has the opportunity to learn a new case, and has to decide whether to retain it or not.

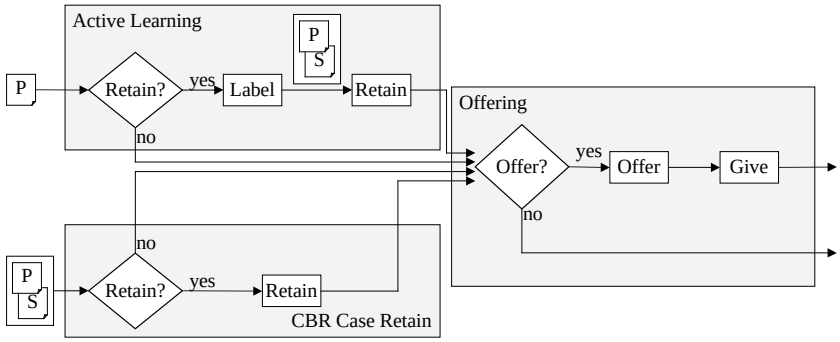

Fig. 1. View of the active learning and CBR case retention decision processes.

As we want to create *case retention strategies* to be used in multiagent scenarios, we divide each full *case retention strategy* in two subprocesses: the individual retention process and the offering process. The individual retention process is responsible of deciding whether a case has to be stored locally or not. The offering process is the responsible of deciding whether to offer a copy of one case or not to some other agents. The way we fulfill each one of these two processes is called a policy. Thus, a retention strategy needs to define an *individual case retention policy* and an *offering policy*. For the *individual case retention policy*, we have seen that there are two different approaches: policies coming from the active learning approach, and policies coming from the CBR case retention approach.

Figure 1 shows the options we have for building a complete case retention strategy. On the left part, we have two different options for the individual case retention policy, and on the right part, we can see the offering policy. Figure 1 also shows the main differences between active learning and CBR case retention: the input to active learning decision policy is a problem with unknown solution, and the input to CBR case retention is a full case (problem plus solution).

In the following sections we will first propose several policies for the individual case retention policy. First we will explain an active learning policy that takes advantage of the fact that the agent is in a multiagent system, and then we will explain several policies from the CBR case retention approach. Finally we will explain with detail the full *case retention strategies* resulting of a combination of all the previous policies with an appropriate offering policy.

3.1 Active Learning Policy

Active learning addresses the problem of deciding if a new example is interesting to be learned or not before knowing its solution. Or alternatively, the question of selecting which are the most interesting examples from a pool of unlabeled examples. But, which criterion can we use to answer these questions?

The answer to the previous question is easier when we restrict to specific learning strategies. For example, if we are learning a concept using the *Version Space* algorithm, we will have three regions in the *Version Space*: a region

containing those problems predicted as positive for all currently consistent versions, a region containing those problems predicted as negative for all currently consistent versions, and the so called "region of uncertainty" [4]. In the region of uncertainty, there are problems that some versions predict as positive and that some versions predict as negative. Clearly, only examples pertaining to the uncertainty region will improve our knowledge. Therefore, a good criterion to assess if a new problem is interesting or not in the version space algorithm is whether this new problem falls in the region of uncertainty or not.

For other kinds of learning algorithms, such as neural networks, analogous criteria can be found, Cohn, Atlas and Ladner [4] define strategies to decide whether a new case will be interesting to retain or not for a neural network classifier. However, finding a general criterion applicable to any learning algorithm is difficult. Seung et al. [12], propose the Shannon information measure of the new example as "suitable guide" for the general case. They also propose the *Query by Committee* (QbC) algorithm as an approximation to this measure.

The QbC algorithm suits our framework particularly well, and can be adapted in a straightforward way to work with lazy learning algorithms in a multiagent system. We propose to use an adapted version of the QbC algorithm presented in [12], that uses all the information available in a group of agents to make the decision. Instead of creating a "virtual committee" as in the original Query by Committee algorithm, we will use a real committee composed of several agents. Thus, we propose the following multiagent version of the Query by committee (MAQbC) algorithm:

1. A new problem P arrives to one agent A_i, and A_i has to decide whether to retain P or not.
2. A_i sends problem P to a set $\mathcal{A}$ of other agents in the system (as in the *Committee* collaboration policy).
3. Each agent $A_j \in \mathcal{A}$ solves the problem and each individual classification is sent back to A_i reified as a SER. The agent A_i also solves the problem individually and stores the obtained SER.
4. A_i builds the set $\mathcal{S}$ containing all the SERs sent by the agents in $\mathcal{A}$ and the SER built by itself. Then, A_i measures the degree of disagreement between the SERs contained in $\mathcal{S}$.
5. If the degree of disagreement is high enough, the new problem P is interesting enough to be retained.

As we will use committees with more than 2 members, we need to measure the degree of disagreement. Let us define d, the degree of disagreement, as follows: $d = V_r/((K-1) * V_w)$, where K is the number of possible solutions, V_w are the votes for the most voted solution and V_r are the votes for the rest of solutions. Notice that when there is a clear majority d approaches 0, and that when the disagreement is maximum (each member of the committee votes for a different class) d approaches 1 (the reason for using $K-1$ instead of K is for normalizing the result between 0 and 1).

In order to decide whether a case is interesting or not, we just have to decide a threshold d_0 for the degree of disagreement d. If $d \geq d_0$ the case is considered interesting, and otherwise it is discarded. This policy will be called the *Informative Disagreement* policy:

- *Informative Disagreement (ID) policy*, each time an agent has the opportunity to retain a new case, the case will be evaluated using the MAQbC algorithm, and only if the degree of disagreement is high enough (i.e. $d \geq d_0$), it will be retained.

Notice that the ID policy is not infallible. In fact, in a situation where most agents err in the same way the degree of disagreement will be low enough, and thus the solution will not be asked to the teacher and the case will not be retained. However, this situation is very unlikely because all or most agents have to give the same erroneous class for the current problem.

3.2 CBR Case Retention Policies

The main difference of the CBR case retention policies with the active learning approach is that CBR policies are able to use the solution of the problem to decide whether to retain the case or not.

Deciding which cases to keep in the case base is one of the main issues in case base maintenance, and most of the work done is focused on deleting cases from the case base because the retrieval time has gone beyond the acceptable limits for an application (*swamping problem*). One of the first policies proposed was random deletion proposed by Markovich and Scott [8]. A more complex approach is taken by Smyth and Keane [14], where a competence preserving case deletion policy is proposed. Leake and Wilson [7] propose a policy to delete cases from the cases base using a performance measure. Finally, Zhu and Yang propose in [15] a technique for selecting a subset of cases from a case base using case addition instead of case deletion. This last technique has the advantage of ensuring that the case base coverage will be above a lower bound. However, these strategies are *off-line* strategies, i.e. they store all the cases in the case base, and after doing that, they analyze the content of the case base for selecting which cases to keep and which cases to delete.

We focus on *on-line* case retention strategies, i.e. each time a new problem arrives, the retention strategy is used to decide whether to retain the new case or not. A more similar approach is taken by Aha et al. [2]. They propose two policies for case retention to be applied into instance based learners in order to reduce the number of cases needed to learn. However, they only focus on centralized systems, and we are dealing with multiagent systems. Our approach can be viewed as a generalization of on-line retention policies where the cases that an individual CBR agent decides not to retain can be nonetheless useful for other CBR agents.

For our experiments, we will use three simple retention policies:

- *On Failure Retain (OFR) policy*, each new case received by the agent will be classified individually. Only if this individual classification differs with the real solution class of the problem, it will be retained.
- *Never Retain (NR) policy*, the agent never retains the new cases.
- *Always (AR) Retain policy*, the agent always retains the new cases.

NR policy represents the scenario where no learning is performed by the learning agent, and AR policy represents the scenario of *passive learning* (i.e. when the learning system retains every new example that has the opportunity to retain). The following section will define complete *case retention strategies* using all the previous retention policies (ID, OFR, NR and AR) and several offering policies.

3.3 Case Retention Strategies

In order to build a complete *case retention* strategy, we need to provide both an *individual case retention* policy and an *offering* policy. In this section, for each possible *individual case retention* policy, we will define one or more *offering* policies to form several full *case retention* strategies.

As Figure 1 shows, the *Offering* process has several steps. First, the agent decides whether to offer the case to other agents or not. After the case is offered, the agent has to decide, from the set of agents that answered positively to the offer, to which agent (or agents) the case is to be sent (In our experiments the agents use the *individual case retention policy* to decide whether they are interested in retaining a case offered by another agent.

Before defining the full case retention strategies, we will define two different scenarios. In the first one, we will consider that there are ownership rights over the cases, and therefore, no copies of the cases can be done. In the second scenario, the agents will be free to make copies of the cases, and therefore, multiple agents will be allowed to retain copies of the same case when need be. We will call the first scenario the *non-copy* scenario, and the second one the *copy* scenario. Several strategies for retaining cases can be defined for each scenario. We will now define the *case retention* strategies applicable in each scenario.

***Non-Copy* Scenario Strategies** In this scenario, we can only allow a single copy of each case in the system. Therefore, only one agent can retain each case. We propose the following strategies for an agent A_i that receives a new problem P to learn:

- *Informative Disagreement - No Offer* strategy (ID-NO): In this strategy, the $MAQbC$ algorithm is used to decide whether to retain P locally or not. The case is never offered to other agents.
- *Informative Disagreement - Offer* strategy (ID-O): The $MAQbC$ algorithm is used to decide whether to retain locally the case or not. If the case is found interesting, A_i will ask the teacher for the correct solution of P. After that, the agent will know which of the agents (including itself) have failed to solve

the problem correctly. If A_i is one of the agents to fail solving P then the case is retained by A_i. Otherwise, as only one agent can retain P, A_i has to choose one of them (currently this selection is done randomly).
- *Never Retain - No Offer* strategy (NR-NO): No retention nor offering process is done. This represents the case of an agent that has no learning capabilities.
- *Always Retain - No Offer* strategy (AR-NO): The new problem is always retained by A_i, and as only a single copy of the case is allowed, no offering can be made.
- *On Failure Retain - No Offer* strategy (OFR-NO): In this strategy, P is only retained by A_i when A_i was not individually capable of solving P correctly. The case is never offered to other agents.
- *On Failure Retain - Offer* strategy (OFR-O): In this strategy, P is only retained by A_i when A_i was not individually capable of solving P correctly. If the case is not retained, it is offered to the other agents. Then, as we are in the *non-copy* scenario, the agent has to choose just one of the agents that have answered requesting P to send only one copy of it. In our experiments, this selection is done randomly.

***Copy* Scenario Strategies** All the previous strategies are applicable to this scenario. For this reason, we will only explain here the different strategies that can only be applied to the *copy* scenario:

- *Informative Disagreement - Offer* (ID-O-copy): The difference of this strategy from *ID-O* is that, as we are in the *copy* scenario, no selection process must be done, and all the agents that have failed to solve P can retain it. Therefore, this strategy works as follows: the *MAQbC* algorithm is applied to decide whether to retain P or not. If the case should be retained, after asking the solution of P from a teacher, all the agents that have not correctly solved P receive a copy of it.
- *On Failure Retain - Offer* (OFR-O-copy): A_i retains the case only when A_i was not individually capable of solving P correctly. Then, P is also offered to the other agents. A copy of the case is sent to each agent that answers requesting a copy. Notice that this is possible only because we are now in the *copy* scenario.

There is another combination of policies that generates a new strategy: *Always Retain - Offer* strategy, where the cases are always retained by every agent. However, this is not an interesting strategy because all the agents in the system will have access exactly to the same cases and will retain all of them.

4 Experimental results

In this section we want to compare the classification accuracy of the *Committee* collaboration policy using all the strategies presented in this chapter. We also present results concerning the resulting size of the case bases.

We use the marine sponge classification problem as our test bed. We have designed an experimental suite with a case base of 280 marine sponges pertaining to three different orders of the *Demospongiae* class (*Astrophorida*, *Hadromerida* and *Axinellida*). In each experimental run the whole collection of cases is divided in two sets, a training set (containing a 10% of the cases), and a test set (containing a 90% of the cases). The training set is distributed among the agents, and then incremental learning is performed with the test set. Each problem in the test set arrives randomly to one agent in the $\mathcal{M}$AC. The goal of the agent receiving a problem is to identify the correct biological order given the description of a new sponge. Once an agent has received a problem, the *Committee* collaboration policy will be used to obtain the identification. Since our experiments use supervised learning, after the committee has solved the problem, there is a supervisor that tells the agent receiver of the problem which was the correct solution. After that, the retention policy is applied. Each agent applies the nearest neighbor rule to solve the problems. The results presented here are the average of 50 experimental runs.

As these experiments try to evaluate the effectiveness of the collaborative learning policies, it is important that the agents really have an incentive to collaborate. If every agent receives a representative (not biased) sample of the data, they will have a lower incentive to ask for cases to other agents since they already have a good sample. For this reason, for experimentation purposes, the agents do not receive the problems randomly. We force biased case bases in every agent by increasing the probability of each agent to receive cases of some classes and decreasing the probability to receive cases of some other classes. This is done both in the training phase and in the test phase. Therefore, each agent will have a biased view of the data.

4.1 Accuracy Comparison

Figures 2 and 3 show the learning curves for two multiagent systems using all the retain strategies that we have presented. For each multiagent system, 8 strategies have been tested: *NR-NO*, *AR-NO*, *OFR-NO*, *ID-NO*, *OFR-O*, *ID-O*, *OFR-O-copy* and *ID-O-copy*. The figures show the learning curve for each strategy. The horizontal axis of the figures represents the number of problems that the agents have received of the test set. The baseline for comparison is the *NR-NO* strategy, where the agents do not retain any cases, and therefore (as we can see in the figures) they do not learn, resulting in an horizontal learning curve around an accuracy of about 50% in all the settings. This is because the training set is extremely small, containing just 28 cases to be distributed between the agents (The *Committee* collaboration policy has proven to obtain results above 88% in this dataset when the agents have a reasonable number of cases [9]). For the experiments that use the *ID* policy, we have set the parameter $d_0 = 0.3$ for the 5 agent scenario and $d_0 = 0.25$ for the 8 agent scenario.

Considering the other seven strategies we can see that they fall in two groups. The first one containing all the non-offering strategies (*AR-NO*, *OFR-NO* and *ID-NO*) and the second one containing all the offering policies (*OFR-O*, *ID-O*,

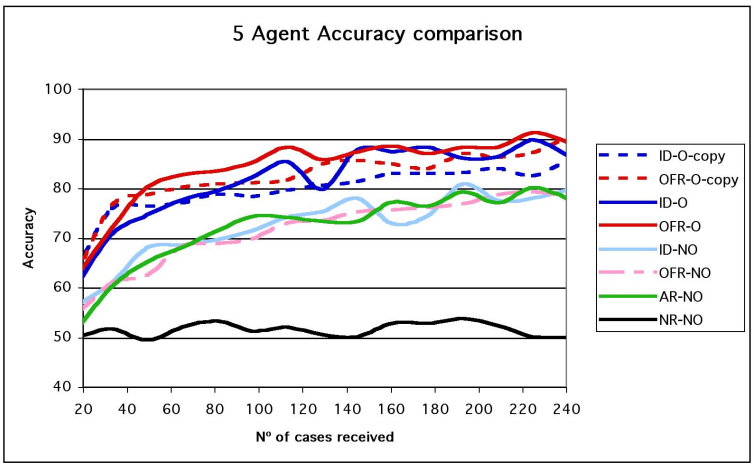

Fig. 2. Accuracy comparison for a $\mathcal{M}$AC system composed of 5 agents.

OFR-O-copy and *ID-O-copy*). Notice also that all the strategies in the offering group have higher accuracies than the strategies in the non-offering group. Let us now analyze them in turn.

Concerning the non-offering strategies, in the 5 agent scenario (Figure 2), they are practically indistinguishable in terms of classification accuracy. They all start with an accuracy of about 55% and reach (after receiving all the cases of the test set) an accuracy of about 80%. If we look at the 8 agent scenario on Figure 3, we can see that we have nearly the same situation. All the non-offering policies start with an accuracy of about 53% and end with an accuracy of about 82%. In this scenario the *ID-NO* strategy seems to work a little better than *AR-NO* and *OFR-NO*. Summarizing, we can say that they all are significantly better than the *NR-NO* strategy but to distinguish between them we have to take in consideration more factors than only accuracy (see section 4.2).

Concerning the group of offering strategies, both Figures 2 and 3 show that the *non-copy* strategies obtain higher accuracies than their respective *copy* scenario versions (i.e. *OFR-O* obtains higher accuracies than *OFR-O-copy* and *ID-O* obtains higher accuracies than *ID-O-copy*). Concerning the *copy* strategies, *OFR-O-copy* obtains higher accuracies than *ID-O-copy*. This difference is not so clear with the *non-copy* strategies, because in the 5 agent scenario *OFR-copy* obtains higher accuracies than *ID-O*, but they both reach accuracies of about 89% in the 8 agent scenario. Summarizing, we can say that *OFR-O* is slightly better than *ID-O* and that both are clearly better than their *copy* scenario versions. The explanation is that if we allow multiple copies of a case in the system, we are increasing the error correlation between the agents. Moreover, the "ensemble effect" [6] states that the combination of uncorrelated classifiers has better results that the combination of correlated ones; increased correlation is the cause of *OFR-O-copy* and *ID-O-copy* achieving lower accuracies than *OFR-O* and *ID-O*.

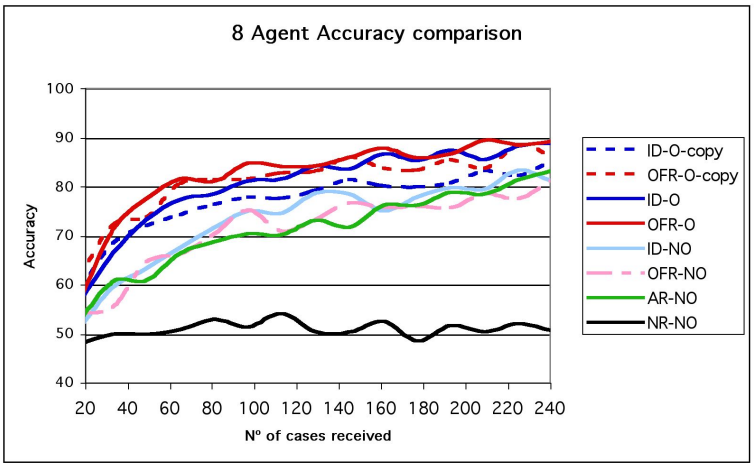

Fig. 3. Accuracy comparison for a $\mathcal{M}$AC system composed of 8 agents.

	5 Agents	*8 Agents*
ID-O-copy	35.93	30.81
OFR-O-copy	56.81	56.18
ID-O	23.96	17.65
OFR-O	34.26	25.70
ID-NO	39.21	28.24
OFR-NO	16.09	11.00
AR-NO	56.00	35.00
NR-NO	5.60	3.50

Table 1. Average case base size of each agent at the end of the learning process.

Comparing both groups of strategies, all the offering strategies obtain always higher accuracies than all the non-offering strategies. Therefore, we can conclude that it is always better for the *Committee* collaboration policy that the agents offer cases to the other agents; the reason is that cases not interesting for some agents can be found interesting by some other agents. In other words, collaboration is better than non collaboration for the retention policies.

4.2 Case Base Size Comparison

Table 1 shows the average size of each individual case base at the end of the learning process (i.e. when all the 252 cases of the test set have been sent to the agents). In all the experiments the size of the initial case base (distributed among the agents) is just 28 cases (the training set). When the agents use the *NR-NO* strategy, since they do not retain any new cases, they just keep the initial cases. For instance, we can see in the 5 agents scenario (where the agents

have in average a case base of 5.60 cases) that 5 times 5.60 is exactly 28 — exactly the number of cases in the training set.

Comparing the case base sizes reached by the non offering strategies (*AR-NO*, *OFR-NO* and *ID-NO*) that achieved nearly indistinguishable accuracies, we can see that there is a great difference among their case base sizes. The strategy that obtained smaller case base sizes was *OFR-NO*, with 16.09 average cases per case base in the 5 agent scenario, and 11.00 average cases per case base in the 8 agent scenario. The next one is *ID-NO*, and the one that obtained the biggest case base sizes was *AR-NO*. Thus, *OFR-NO* is better than the other two, because has the same accuracy but with a smaller case base size. However, we have to take into consideration that the *ID-NO* strategy uses less information, because it doesn't need to ask the solution of the problems before deciding whether to retain them or not.

In the case of the offering strategies, the strategies working in the *non-copy* scenario obtain smaller case base sizes than the strategies working in the *copy* scenario. This is not surprising, because in the *copy* scenario we allow multiple copies of each case to be retained, thus increasing the amount of cases retained by the agents. Since they also obtain better accuracies this result still reinforces the fact that the strategies working in the *non-copy* scenario obtain better results than the strategies working in the *copy* scenario. These strategies obtain higher accuracies because the error correlation is lower and then the ensemble effect is stronger.

Comparing the strategies based in active learning (*ID-O* and *ID-O-copy*) with the strategies based in CBR case retention (*OFR-O* and *OFR-O-copy*), we see that active learning strategies obtain smaller case base sizes. However, we are unable to say that *ID-O-copy* is better than *OFR-O-copy* because *OFR-O-copy* obtained higher classification accuracies. Comparing *ID-O* with *OFR-O*, we can see that *ID-O* obtains smaller case base sizes, but *OFR-O* obtained slightly higher accuracies.

5 Conclusions

As we have seen, some retention strategies for Ensemble CBR are clearly better than others. For instance, offering strategies always obtain greater case base sizes than non-offering strategies. But this increase in case base size is highly justified by a large increase in classification accuracy. Therefore, we can conclude that for $\mathcal{M}$AC systems using the *committee* policy it is better for an agent to offer cases to other agents.

Strategies working in the *copy* scenario have an increased case base size with a lower accuracy. Therefore they are clearly worse than the *non-copy* strategies. This may seem not intuitive, but it's an expected issue of the "ensemble effect" since copied cases increase the error correlation between agents.

Comparing strategies coming from active learning and strategies coming from CBR case retention, there is no clear winner. Strategies based on CBR case retain (*OFR-NO*, *OFR-O* and *OFR-O-copy*) usually obtain higher accuracies,

but strategies based on active learning (*ID-NO*, *ID-O* and *ID-O-copy*) obtain smaller case bases (except for *ID-NO*, that obtains relatively large case bases). Moreover, while comparing active learning strategies with CBR case retention strategies we have to take into consideration more factors. The first one is that active learning strategies do not need to ask for the right solution for every problem before deciding whether to retain it or not. Thus, they can avoid a lot of questions to the teacher. Moreover, we also have to take into consideration that the Informative Disagreement (*ID*) policy has to be tuned to each system with the adequate threshold d_0. In our experiments we have found that this is very easy when the committee has a not too small number of members. However, when the committee is very small (for instance we have experimented with a committee of 3 agents), it's not possible to find a d_0 that obtains good results, because there are very few possible values for the degree of disagreement d.

At the beginning of the experiments section we have said that, in order to give the agents an incentive to collaborate, we have forced biased case bases in every agent by increasing the probability of each agent to receive cases of some classes and decreasing the probability to receive cases of some other classes. To see the effect of retention policies under unbiased conditions, we have performed all the experiments presented in this section but randomly distributing the cases in the agents rather than forcing biased case bases. We found that the difference between offering and non-offering strategies is smaller than in the scenario of biased case bases, but that offering strategies still improve the performance. This result was expected, since when the agents receive the problems randomly, each agent receives a representative sample of examples, reducing the need of obtaining cases from other agents. In the unbiased scenario both non-offering strategies (*OFR-NO* and *AR-NO*) obtain slightly (but significantly) lower accuracies than *OFR-O*, and the same happens with *ID-NO* respect to *ID-O*. The reason for the small increment in accuracy of the offering policies (*OFR-O* and *ID-O*) is that when agents use offering policies, all the agents in the system have access to a greater amount of cases, and thus they retain more cases on average, leading to a slightly higher individual accuracy. This increment in individual accuracy is the cause of the small increment in accuracy of the offering policies in the unbiased scenario.

Summarizing, there are two main conclusions. The first one is that for case retention cooperating with other agents is always beneficial, since the offering policies always perform better than non-offering policies and *committee* collaboration policy works better than individual problem solving. The second conclusion is that working in the *non-copy* scenario to avoid error correlation between the agents' case bases is preferable, since we can obtain more benefits from the *committee* collaboration policy. This second result is quite interesting, because it could seem intuitive to think that the more cases each agent is allowed to retain, the greater classification accuracy the committee will obtain. Moreover, the restriction introduced by the *non-copy* scenario (only one copy of each case allowed in the system) could seem arbitrary, but the reduction of the error correlation between the agents obtained has a strong enough effect to prefer it to

the *copy* scenario. It remains as future work to explore intermediate scenarios between the *copy* and *non-copy*, to see if a better tradeoff between individual accuracy and error correlation can be found.

Finally, notice that the ID policy (used in active learning) takes into account information obtained from other agents. As a future work, we plan to develop new CBR case retention policies that also take advantage of information obtained from other agents.

Acknowledgements The authors thank Josep-Lluís Arcos of the IIIA-CSIC for his support and for the development of the Noos agent platform. Support for this work came from CIRIT FI/FAP 2001 grant and projects TIC2000-1414 "eInstitutor" and SAMAP (MCYT-FEDER) TIC2002-04146-C05-01.

References

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications*, 7(1):39–59, 1994.

[2] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.

[3] Steven J. Brams and Peter C. Fishburn. *Approval Voting*. Birkhauser, 1983.

[4] David A. Cohn, Les Atlas, and Richard E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[5] L. K. Hansen and P. Salamon. Neural networks ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:993–1001, 1990.

[6] Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 231–238. The MIT Press, 1995.

[7] David B. Leake and David C. Wilson. Remembering why to remember: Performance-guided case-base maintenance. In *EWCBR-2000*, LNAI, pages 161–172. Springer Verlag, 2000.

[8] S. Markovich and P. Scott. The role of forgetting in learning. In *ICML-88*, pages 459–465. Morgan Kaufman, 1988.

[9] S. Ontañón and E. Plaza. Learning when to collaborate among learning agents. In *ECML-2001*, LNAI, pages 394–405. Springer-Verlag, 2001.

[10] Santiago Ontañón and Enric Plaza. Learning to form dynamic committees. In *Int. Conf. Autonomous Agents and Multiagent Systems AAMAS'03*, 2003.

[11] Enric Plaza and Santiago Ontañón. Ensemble case-based reasoning: Collaboration policies for multiagent cooperative cbr. In I. Watson and Q. Yang, editors, *In Case-Based Reasoning Research and Development: ICCBR-2001*, number 2080 in LNAI, pages 437–451. Springer-Verlag, 2001.

[12] H. S. Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Computational Learing Theory*, pages 287–294, 1992.

[13] B. Smyth. The utility problem analysed: A case-based reasoning persepctive. In *EWCBR-96*, LNAI, pages 234–248. Springer Verlag, 1996.

[14] Barry Smyth and Mark T. Keane. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In *IJCAI-95*, pages 377–382, 1995.

[15] Jun Zhu and Qiang Yang. Remembering to add: Competence-preserving case-addition policies for case base maintenance. In *IJCAI-99*, pages 234–241, 1999.