Justification-based Multiagent Learning

Santiago Ontañón SANTI@IIIA.CSIC.ES
Enric Plaza ENRIC@IIIA.CSIC.ES
IIIA, Artificial Intelligence Research Institute
CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia (Spain)

Abstract

Committees of classifiers with learning capabilities have good performance in a variety of domains. We focus on committees of agents with learning capabilities where no agent is omniscient but has a local, limited, individual view of data. In this framework, a major issue is how to integrate the individual results in an overall result—usually a voting mechanism is used. We propose a setting where agents can express a symbolic justification of their individual results. Justifications can then be examined by other agents and accepted or found wanting. We propose a specific interaction protocol that supports revision of justifications created by different agents. Finally, the opinions of individual agents are aggregated into a global outcome using a weighted voting scheme.

1. Introduction

Lazy learning can provide multiagent systems with the capability to autonomously learn from experience. We present a framework where agents that use lazy learning can collaborate in order to solve classification tasks. All the agents in our systems are able to solve the problems individually, but the incentive for collaboration is to improve the classification accuracy. The main topic here is how to aggregate the predictions coming from different agents into a global prediction. We propose a multiagent collaboration scheme called the *Justification Endorsed Collaboration* policy in which the agents provide a symbolic justification of their individual results. These justifications can be then examined by other agents in the system to assess a measure of confidence on each individual result. The individual results are finally aggregated by means of a weighted voting scheme that uses these computed confidence measures as weights.

A strongly related area is the field of multiple model learning (also known as multiclassifier systems, ensemble learning, or committees of classifiers). A general result in multiple model learning (Hansen & Salamon, 1990) proved that combining the results of multiple classifiers gives better results than having a single classifier. Specifically, when the individual classifiers make uncorrelated errors, and have an error rate lower than 0.5, the combined error rate must be lower than the one made by the best of the individual classifiers. The BEM (*Basic Ensemble Method*) is presented in (Perrone & Cooper, 1993) as a basic way to combine continuous estimators by taking the average of all the predictions. Since then, many methods have been proposed: *Cascade Generalization* (Gama, 1998), where the classifiers are combined in a sequential way; or *Bagging* (Breiman, 1996) and *Boosting* (Freund & Schapire, 1996), where the classifiers are combined in parallel, are good examples.

The main difference of our approach is that ensemble methods such as Bagging or Boosting are centralized methods, that have control over the entire training set, and that they artificially create different classifiers with the only goal of improving classification accuracy. In our approach, we assume that we have several individual agents, and that each one has collected experience on its own. Therefore, we don't have any control over the contents of the individual agents' training sets. Moreover, the agents will keep private their individual data, so no agent can have access to the internal experience of the other agents. In Bagging, for instance, each individual classifier is created starting from a sample of cases of the original training set, but with repetitions, i.e. there is overlapping among the training sets of the individual classifiers,

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

and it is this that enables Bagging to improve the accuracy over a single classifier. However, in our system, the agents do not know if there is some overlapping among the training sets of the individual agents or not. Therefore, the collaboration method to be used has to be strong enough to work with or without overlapping, and whatever the individual training sets are. In the experiments section, we present experimental results showing that our method works well in different scenarios, while standard voting fails in some of them.

The structure of the article is as follows: Section 2 succinctly defines the multiagent systems with which we work. Then, Section 3 defines the learning method that the individual agents use in our system to solve the classification problems and to build the symbolic justifications. Section 4 presents the justification mechanism used by the agents in order to aggregate the individual predictions, and finally Section 5 shows an experimental evaluation of the justification method. The paper closes with the conclusions section.

2. Multiagent Case-Based Learning

Each agent in our systems use case based reasoning (CBR) in order to solve classification tasks, and each individual agent owns a private case base. Therefore, a multiagent system is a collection of pairs: $\{A_i, C_i\}_{i=1...n}$, where C_i is the local case base of agent A_i and n is the number of agents. As we have said, each agent is able to completely solve a problem by its own, but it will collaborate with other agents if this can improve the classification accuracy.

In previous work, we presented the *committee* collaboration policy (Ontañón & Plaza, 2001) for multiagent systems. When solving a problem using this strategy, each agent can vote for one or more solution classes and the final solution is the most voted solution. Using this method, the agents were able to obtain higher classification accuracies than working individually. The *committee* collaboration policy uses a variation of Approval Voting called *Bounded Weighted Approval voting*.

3. Lazy Induction of Descriptions

In this section we describe LID(Armengol & Plaza, 2001), the CBR method used by the agents in our experiments. LID is a CBR method that can provide a symbolic justification of why it has classified a given problem in a specific solution class. We will first present the representation of the cases in LID, then the heuristic measure used for guiding search, and finally the main steps of LID.

LID uses the feature term formalism for representing cases. *Feature Terms* (ψ*-terms*) are a generalization of the first order terms. The main difference is that in first order terms (e.g. $person(barbara, john, dianne)$) the parameters of the terms are identified by position, and in a feature term the parameters (called *features*) are identified by name (e.g. $person[name \doteq barbara, father \doteq john, mother \doteq dianne]$). Another difference is that feature terms have a *sort*, for instance, the previous example belongs to the sort *person*. These sorts can have subsorts (e.g. *man* and *woman* are subsorts of *person*). Feature terms have an informational order relation ($\sqsubseteq$) among them called subsumption, where $\psi \sqsubseteq \psi'$ means all the information contained in ψ is also contained in ψ' (we say that ψ subsumes ψ'). The minimal element is $\perp$ and is called *any*, that represents the minimum information. All the feature terms of other sorts are subsumed by *any*. When a feature term has no features (or all of its features are equal to $\perp$) it is called a *leaf*. A *path* $\rho(\psi, f_i)$ is defined as a sequence of features going from the term ψ to the feature f_i.

Figure 1 shows a graphical representation of a feature term. Each box in the figure represents a node. On the top of the boxes the sort of the nodes is shown, and on the lower part, all the features that have a value different than *any* are shown. The arrows mean that the feature on the left part of the arrow takes the node on the right as value. In Figure 1 the two nodes labelled with *No* and the *Tylostyle* node are leaf nodes. The path from the root to the *Tylostyle* node in the example is *Spiculate-Skeleton.Megascleres.Smooth-form*.

LID only considers the *leaf* features of the terms and uses a heuristic measure to decide which are the most discriminatory *leaf* features contained in a problem description. The heuristic used is the minimization of the RLM distance (López de Mántaras, 1991). The RLM distance assesses how similar are two partitions over a set of cases (less distance, more similarity). Given a feature f, its possible values induce a partition Π_f over the set of cases. Therefore, we can measure the distance between the correct partition Π_c (given by the solution classes) and the partition induced by a feature. We say that a feature f is *more discriminatory* than the feature f' if $RLM(\Pi_f, \Pi_c) < RLM(\Pi_{f'}, \Pi_c)$. In other words, a more discriminatory feature classifies the cases in a more similar way to the correct classification of cases.

LID uses a top-down heuristic strategy to build a symbolic description $\mathcal{D}$ for a problem P. The partial de-

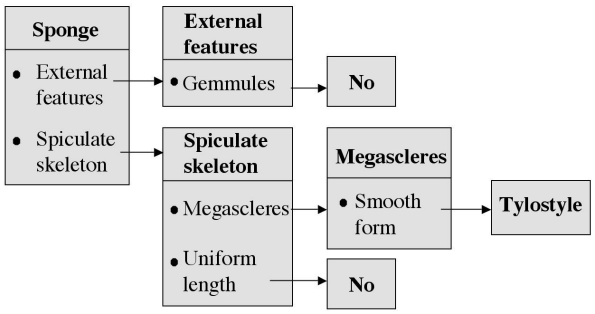

Figure 1. Example of symbolic justification returned by LID in the marine sponge classification problem.

Function LID $(\mathcal{S}_{\mathcal{D}_i}, P, \mathcal{D}_i, \mathcal{K})$
  if stopping-condition$(\mathcal{S}_{\mathcal{D}_i})$
    then return $(\mathcal{S}_{\mathcal{D}_i}, \mathcal{D}_i, K_{\mathcal{D}_i})$
    else $f_d :=$ Select-leaf $(P, \mathcal{S}_{\mathcal{D}_i})$
      $\mathcal{D}_{i+1} :=$ Add-path$(\rho(P, f_d), \mathcal{D}_i)$
      $\mathcal{S}_{\mathcal{D}_{i+1}} := \{x \in \mathcal{S}_{\mathcal{D}_i} |\ \mathcal{D}_{i+1} \sqsubseteq x\}$
      LID $(\mathcal{S}_{\mathcal{D}_{i+1}}, P, \mathcal{D}_{i+1}, \mathcal{K})$
  end-if
end-function

Figure 2. The LID algorithm. $\mathcal{D}_i$ is the similitude term, $\mathcal{S}_{\mathcal{D}_i}$ is the discriminatory set of $\mathcal{D}_i$, K is the set of solution classes. *Select-leaf* uses the RLM distance to select the most discriminatory feature.

scription $\mathcal{D}$ satisfies the following three conditions: $\mathcal{D}$ subsumes P, $\mathcal{D}$ contains the most discriminatory features of P and $\mathcal{D}$ subsumes a subset of the case base: $\mathcal{S}_{\mathcal{D}}$ (called the *discriminatory set*). We can consider $\mathcal{D}$ as a *similitude term*, i.e. a symbolic description of similarities between a problem P and the retrieved cases $\mathcal{S}_{\mathcal{D}}$.

The main steps of LID are shown in Figure 2. LID initially receives these parameters: $\mathcal{S}_{\mathcal{D}_0}$ is the case base C_i of the agent, the problem P to solve, an initially empty similitude term $\mathcal{D}_0 = \perp$, and the set of solution classes $\mathcal{K}$. The goal of LID is to find the right solution class for the problem P. LID works as follows: the first step is to check the *stopping-condition*, that tests whether all the cases in $\mathcal{S}_{\mathcal{D}_i}$ belong to the same solution class or not. If this stopping condition is not satisfied, LID selects one of the features (the most discriminant feature according to the RLM distance) of the problem P and adds it to the current similitude term $\mathcal{D}_i$ (specializing it with the value found in P) to construct the new similitude term $\mathcal{D}_{i+1}$. Next, LID is recursively called using the new similitude term $\mathcal{D}_{i+1}$ and a new discriminatory set $\mathcal{S}_{\mathcal{D}_{i+1}}$ containing only those cases in $\mathcal{S}_{\mathcal{D}_i}$ subsumed by $\mathcal{D}_{i+1}$. This process continues until the similitude term $\mathcal{D}_i$ is specific enough to be able to satisfy the stopping condition or until there are no more possible features to add to $\mathcal{D}_i$.

The output of LID for a problem P is the tuple $\langle \mathcal{S}_{\mathcal{D}}, \mathcal{D}, K_{\mathcal{D}} \rangle$, where $K_{\mathcal{D}}$ is the predicted solution class (or solution classes if LID has not been able to select a single solution class), i.e. $K_{\mathcal{D}} = \{S_k |\ \exists x \in \mathcal{S}_{\mathcal{D}} \wedge solution(x) = S_k\}$. As the similitude term $\mathcal{D}$ contains the common information between the retrieved cases (discriminatory set) and the problem P (the relevant information that LID has used to classify the problem into the predicted solution class), $\mathcal{D}$ will be used as the symbolic justification for having classified the problem P in the class (or classes) $K_{\mathcal{D}}$.

Example: Let us illustrate the LID method with an example. Imagine that we have a case base of 252 cases of the marine sponge identification problem. Each sponge can belong to one of three classes: *Astrophorida*, *Hadromerida* or *Axinellida* and we have a problem P to solve. Let's call the initial similitude term $\mathcal{D}_0 = \perp$. The first discriminatory set $\mathcal{S}_{\mathcal{D}_0}$ will contain the entire case base of 252 cases (95 *Astrophorida* cases, 117 *Hadromerida* cases and 68 *Axinellida* cases).

The first *leaf* feature selected by LID as the most discriminatory feature according to the RLM distance is the feature identified by the path "*External-features.Gemmules*". This path has the value "*No*" in the problem P, so the path "*External-features.Gemmules*" with the value "*No*" is added to the similitude term $\mathcal{D}_0$ to build the new similitude term $\mathcal{D}_1$ as we see in Figure 1 (that shows the similitude term returned by LID at the end of execution). The new discriminatory set $\mathcal{S}_{\mathcal{D}_1}$ contains now only those cases subsumed by $\mathcal{D}_1$ (i.e. all those cases from $\mathcal{S}_{\mathcal{D}_0}$ having the value "*No*" in the feature "*External-features.Gemmules*"): 16 *Astrophorida* cases, 35 *Hadromerida* cases and 8 *Axinellida* cases.

The next path selected by LID is "*Spiculate-skeleton.Megascleres.Smooth-form*" which has the value *Tylostyle* in the problem P, so this path with this value is added to the similitude term $\mathcal{D}_1$ to build the new similitude term $\mathcal{D}_2$. The new discriminatory set $\mathcal{S}_{\mathcal{D}_2}$ contains now only 5 *Astrophorida* cases and 25 *Hadromerida* cases.

LID now selects the path "*Spiculate-skeleton.Uniform-length*" with value "*No*", which is also added to $\mathcal{D}_2$ to build the new similitude term $\mathcal{D}_3$. The new discriminatory set $\mathcal{S}_{\mathcal{D}_3}$ now contains only 25 *Hadromerida* cases.

Finally, the stopping condition is met because the discriminatory set $\mathcal{S}_{\mathcal{D}_3}$ contains cases with only one so-

lution class. Therefore, LID will return the similitude term $\mathcal{D}_3$, the discriminatory set $\mathcal{S}_{\mathcal{D}_3}$ and the solution class $K_{\mathcal{D}} = \{Hadromerida\}$. The similitude term can be interpreted as a *justification* for having classified the problem P into the class *Hadromerida*, i.e. "The problem P belongs to the class *Hadromerida* because it has no *Gemmules*, the *spiculate skeleton* does not have a uniform length and the *megascleres* (in the *spiculate skeleton*) have a *tylostyle smooth form*". The discriminatory set returned by LID is the set of cases that *endorse* the justification.

4. Collaborative Learning

In the *committee* collaboration policy mentioned in Section 2, the agents aggregated all the solutions obtained individually through a voting process. However, this is not always the best way to aggregate the information. In this section we present a new collaboration policy (the *Justification Endorsed Collaboration* (JEC) policy) in which each agent can give a symbolic justification of its individual solution. In this way, the solutions that are not endorsed by good justification will not have the same strength in the voting process as the solutions that are well endorsed.

To create a justification of the individual solution for a problem P, an agent solves P individually using LID. The answer of LID is a tuple $\langle \mathcal{S}_{\mathcal{D}}, \mathcal{D}, K_{\mathcal{D}} \rangle$. The symbolic similitude term $\mathcal{D}$ is used as the justification for the answer. Specifically, the agent builds a *justified endorsement record*:

Definition: A *justified endorsement record* (JER) $\mathbf{J}$ is a tuple $\langle S, \mathcal{D}, P, A \rangle$, where S is the solution class for problem P, $\mathcal{D}$ is the similitude term given by LID i.e. the symbolic justification of the classification of P in S, and A is the agent creator of the record.

We will use the dot notation to denote an element in the tuple (i.e. $\mathbf{J}.\mathcal{D}$ is the similitude term $\mathcal{D}$ in $\mathbf{J}$).

Each JER contains the justification that endorses one solution class as the possible correct solution for a problem P. When the output of LID contains more than one possible solution class several JERs are built. We will denote by $\mathbf{J}_{A_j}(P)$ the set of JERs built by A_j for a problem P. An agent will generate as many JERs as solution classes contained in the set $K_{\mathcal{D}}$ given by LID. As we will see later, these JERs are sent to the other agents in the system for examination. When an agent receives a JER to examine, the justification contained in the JER is contrasted against the local case base. To examine a justification $\mathbf{J}$, an agent obtains the set of cases contained in its local case base that are subsumed by $\mathbf{J}.\mathcal{D}$. The more of these cases that belong to the same solution class $\mathbf{J}.S$ predicted by $\mathbf{J}$, the more positive the examination will be. All the information obtained during the examination process of a JER by an agent is stored in an *examination record*:

Definition: An *examination record* (XER) $\mathbf{X}$ is a tuple $\langle \mathbf{J}, Y_{\mathbf{J}}^{A_j}, N_{\mathbf{J}}^{A_j}, A_j \rangle$, where $\mathbf{J}$ is a JER, $Y_{\mathbf{J}}^{A_j} = |\{x \in C_j |\ \mathbf{J}.\mathcal{D} \sqsubseteq x \wedge \mathbf{J}.S = solution(x)\}|$ is the number of cases in the agent's case base *subsumed* by the justification $\mathbf{J}.\mathcal{D}$ that belong to the solution class $\mathbf{J}.S$ proposed by $\mathbf{J}$, $N_{\mathbf{J}}^{A_j} = |\{x \in C_j |\ \mathbf{J}.\mathcal{D} \sqsubseteq x \wedge \mathbf{J}.S \neq solution(x)\}|$ is the number of cases in the agent's case base *subsumed* by justification $\mathbf{J}.\mathcal{D}$ that *do not* belong to that solution class, and A is the agent that has created the examination record.

As we will see, those examination records are used to create a *confidence* measure about each individual solution. Then, using these confidence measures, the individual solutions are aggregated to obtain the final prediction. In the next section, the JEC policy is explained in detail.

4.1. Justification Endorsed Collaboration Policy

When an agent A_i wants to solve a problem P using the justification endorsed collaboration policy, the procedure followed consists of:

1. A_i (called the *convener agent*) broadcasts problem P to all agents in the multiagent system,
2. each agent A_j in the system solves the problem P and builds a set of the JERs using LID: $\mathbf{J}_{A_j}(P)$,
3. each agent A_j broadcasts its justified endorsement records $\mathbf{J}_{A_j}(P)$ to all agent members of the system,
4. every agent A_j of the system now has a collection of justified endorsement records $\mathbf{J}(P) = \bigcup_{a=1\ldots n} \mathbf{J}_{A_a}(P)$ (notice that all the JERs built by every agent are contained in $\mathbf{J}(P)$ including their own JERs);
   (a) for each JER $\mathbf{J} \in \mathbf{J}(P)$ (including their own) agent A_j examines the justification $\mathbf{J}.\mathcal{D}$ of the JER against her case base,
   (b) after contrasting the justification against the local case base, A_j builds a XER $\mathbf{X}_{\mathbf{J}} = \langle \mathbf{J}, Y_{\mathbf{J}}^{A_j}, N_{\mathbf{J}}^{A_j}, A_j \rangle$ for each JER,
   (c) every agent A_j sends the *examination record* built for each justification to the convener agent A_i,

5. the convener agent A_i now has the set $\mathbf{X}(P)$ containing all the examination records for each JER sent by all other agents as result of examining the justifications of the member agents against each individual case base (including all the ones by A_i itself);

   (a) for each justification $\mathbf{J}$ a *confidence* estimate is computed from the relevant examination records $\{\mathbf{X} \in \mathbf{X}(P) | \ \mathbf{X}.\mathbf{J} = \mathbf{J}\}$. The confidence estimate $\mathsf{C}(\mathbf{J})$ is computed as the ratio between the sum of the positive cases and the sum of positive and negative cases recorded in the examination records for that justification:

$$\mathsf{C}(\mathbf{J}) = \frac{\sum_{a=1\ldots n} Y_{\mathbf{J}}^{A_a}}{\sum_{a=1\ldots n} Y_{\mathbf{J}}^{A_a} + N_{\mathbf{J}}^{A_a}}$$

   (b) agent A_i can now perform a weighted voting scheme on the solution classes. For each solution class S_k, agent A_i collects all justifications for that class and adds the confidence estimates: $\mathsf{C}(S_k) = \sum_{\mathbf{J} \in \{\mathbf{J}(P) | \mathbf{J}.S = S_k\}} \mathsf{C}(\mathbf{J})$,

   (c) the solution selected is the one with greatest confidence $\mathbf{S}(P) = argmax_{k=1\ldots K}(\mathsf{C}(\mathsf{S_k}))$.

Of course, we could give a normalized class confidence by computing the ratio between the sum of positive cases and the sum of positive and negative cases of all relevant examination records, but we are only interested in the partial order that tells us which solution is the one with greatest confidence.

Notice that the examination process of the JERs is not sensitive to the distribution of cases among the agents. In other words, given a JER $\mathbf{J}$, the confidence measure $\mathsf{C}(\mathbf{J})$ that the convener agent computes will not change if we completely redistribute the cases among the agents. $\mathsf{C}(\mathbf{J})$ also remains unchanged if we change the number of agents. A demonstration can be found in (Ontañón & Plaza, 2003). This ensures that the confidence measures computed are robust and that they are always computed taking advantage of all the available information in the system. However, the confidence estimated depend on the degree of redundancy —i.e. if we duplicate a case in the system, the confidence estimates will vary.

Example: Let us illustrate how the process works with an example. Imagine a system composed of 3 agents: A_1, A_2 and A_3. The agent A_1 wants to solve the problem P of marine sponge classification (thus, A_1 will play the role of *convener* agent). First, A_1 sends the problem P to A_2 and A_3. All three agents try to solve the problem individually using LID.

After solving the problem P using LID, the agent A_1 has found that the solution class for the problem P is *Hadromerida* and the justification is the one shown in Figure 1. Therefore, A_1 builds a justified endorsing record $\mathbf{J}_1 = \langle Hadromerida, \mathcal{D}_1, P, A_1 \rangle$, where $\mathcal{D}_1$ is the justification shown in Figure 1. Then, $\mathbf{J}_1$ is sent to A_2 and A_3. Analogously, A_2 builds the JER $\mathbf{J}_2 = \langle Axinellida, \mathcal{D}_2, P, A_2 \rangle$ and A_3 builds the JER $\mathbf{J}_3 = \langle Hadromerida, \mathcal{D}_3, P, A_3 \rangle$, and sends them to the other agents (notice that each agent could generate more than one JER if $\mathcal{D}_i$ covers cases in more than one solution class).

At this point, each agent (A_1, A_2 and A_3) has the set of JERs $\mathbf{J}(P) = \{\mathbf{J}_1, \mathbf{J}_2, \mathbf{J}_3\}$, containing all the JERs built by all the agents. It's time to build the examination records.

For instance, when A_2 starts examining the JER $\mathbf{J}_1$ coming from the agent A_1, all the cases in the case base of A_2 that are subsumed by the justification $\mathbf{J}_1.\mathcal{D}$ are retrieved: 13 cases, 8 cases belonging the *Hadromerida* solution class, and 5 cases belonging to *Astrophorida*. That means that A_2 knows 5 cases that completely satisfy the justification given by A_1, but that do not belong to the *Hadromerida* class. Therefore, the XER built for the justification $\mathbf{J}_1$ is $\mathbf{X}_1 = \langle \mathbf{J}_1, 8, 5, A_2 \rangle$. Then, A_2 continues by examinating the next JER, $\mathbf{J}_2$ (its own JER), and finds only 3 cases belonging the *Axinellida* class. Therefore, A_2 builds the following XER: $\mathbf{X}_2 = \langle \mathbf{J}_2, 3, 0, A_2 \rangle$. Finally, A_2 also builds the XER for the JER $\mathbf{J}_3$: $\mathbf{X}_3 = \langle \mathbf{J}_3, 5, 1, A_2 \rangle$. Those three XERs are sent to the convener agent A_1. In the same way, A_3 also builds its own XERs and sends them to A_1. The convener agent A_1 also builds its own XERs and stores them.

After having received the rest of examination records: $\mathbf{X}_4 = \langle \mathbf{J}_1, 7, 0, A_3 \rangle$, $\mathbf{X}_5 = \langle \mathbf{J}_2, 2, 5, A_3 \rangle$, $\mathbf{X}_6 = \langle \mathbf{J}_3, 10, 0, A_3 \rangle$, $\mathbf{X}_7 = \langle \mathbf{J}_1, 15, 0, A_1 \rangle$, $\mathbf{X}_8 = \langle \mathbf{J}_2, 1, 4, A_1 \rangle$, $\mathbf{X}_9 = \langle \mathbf{J}_3, 6, 1, A_1 \rangle$, A_1 builds the set $\mathbf{X}(P) = \{\mathbf{X}_1, ..., \mathbf{X}_9\}$. Then, A_1 computes the confidence measures for each JER. For the JER $\mathbf{J}_1$, the confidence measure will be obtained from the XERs $\mathbf{X}_1$, $\mathbf{X}_4$ and $\mathbf{X}_7$ (the XERs that refer to $\mathbf{J}_1$): $\mathsf{C}(\mathbf{J}_1) = (8+7+15)/(8+5+7+0+15+0) = 0.85$. In the same way, the confidence $\mathsf{C}(\mathbf{J}_2) = 0.40$ will be computed from $\mathbf{X}_2$, $\mathbf{X}_5$ and $\mathbf{X}_8$ and the confidence $\mathsf{C}(\mathbf{J}_3) = 0.91$ from $\mathbf{X}_3$, $\mathbf{X}_6$ and $\mathbf{X}_9$. Notice how the weakest JER ($\mathbf{J}_2$) has obtained the lowest confidence, while stronger justifications obtain higher confidence values.

Once all the confidence measures of the JERs have

been computed they can be aggregated to obtain the confidences of the solution classes. For instance, there are two JERs ($\mathbf{J}_1$ and $\mathbf{J}_2$) endorsing *Hadromerida* as the solution, and one JER ($\mathbf{J}_2$) endorsing *Axinellida* as the solution. Therefore, the confidence measures for the solution classes are: $\mathsf{C}(Hadromerida) = 0.85 + 0.91 = 1.76$ and $\mathsf{C}(Axinellida) = 0.40$. The selected solution class is *Hadromerida* because it is the one with highest confidence.

5. Experimental Results

In this section we present experimental results showing the benefits of using the JEC policy versus the standard committee collaboration policy. We use the marine sponge classification problem as our testbed. Sponge classification is interesting because the difficulties arise from the morphological plasticity of the species, and from the incomplete knowledge of many of their biological and cytological features. Sponges have a complex structure, making them amenable to build complex justifications using LID.

In order to compare the performance of the justification based approach and the voting approach, 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 an experimental run, training cases are randomly distributed to the agents. In the testing phase, problems arrive randomly to one of the agents. The goal of the agents is to identify the correct biological order given the description of a new sponge. We have experimented with 5, 7, 9, 10 and 16 agent systems using LID as the classification method. The results presented here are the result of the average of 5 10-fold cross validation runs. For each scenario, the accuracy of both the JEC policy (justification) and the committee are presented. The accuracy of the agents working in isolation is also presented for comparison purposes. In the individual results, the individual accuracy of the convener agent is considered.

In order to test the generality of the technique, three different kind of scenarios will be presented: the *unbiased scenarios*, the *biased scenarios* and the *redundancy scenarios*. In the unbiased scenarios, the training set will be divided into several disjoint subsets in a random way, and each subset will be given to each one of the individual agents. In the biased scenarios, those subsets will not be randomly generated, and the agents will have a skewed view of the problem, i.e. some agents will receive very few (or no) problems of some classes, and will receive lots of problems of some other classes. And finally, in the redundancy scenarios, the subsets of the training set will not be disjoint, i.e. some of the cases will be present in more than one subset.

5.1. Unbiased Scenarios

In the unbiased scenarios, the training set is divided into as many subsets as there are agents in the system. Then, each one of these subsets is given to each one of the individual agents as the training set. For instance, in the 5 agent scenario, the training set of each agent consists of about 50 sponges and in the 16 agent scenario it consists of about 16 sponges. Therefore, in the scenarios with many agents, as the individual case bases are smaller, the individual accuracies will also be lower, leading to a greater incentive to collaborate.

Table 1 shows the classification accuracies obtained by several groups of agents using the JEC policy and the committee collaboration policy. The first thing we notice is that the agents using justifications obtain higher accuracies. This difference is noticeable in all the scenarios, for instance in the 5 agent scenario, the agents using justifications obtained an accuracy of 88.50% and the agents using the committee collaboration policy obtained an accuracy of 88.36%. The difference is not great in systems with few agents, but as the number of agents increase, the accuracy of the agents using the committee policy drops. For instance, in the 16 agents scenario, the accuracy for the agents using justifications is of 87.28% and the accuracy for the committee policy is 85.71%. For testing whether this effect prevails when increasing even more the number of agents, we have performed an experiment with 25 agents (where each agent have only about 10 cases and the individual accuracy is 58.21%), and the result is that the agents using justifications still obtained an accuracy of about 87.28% while the agents using the committee collaboration strategy obtained an accuracy of 84.14%. These results show that even when the data is very fragmented, the confidence measures (that are not affected by the fragmentation of the data) computed by the agents are able to select which of the individual solutions are better based on the justification given.

Table 1 also shows the accuracy obtained by the agents when they solve the problems individually (without collaborating with the other agents). The table shows that as the number of agents in the experiments increases (and the size of the individual training sets diminishes) the individual accuracy drops fast. For instance, in the 16 agents scenario, the individual accuracy is only 67.07% while the accuracy obtained in the same system using the JEC policy is 87.28%. For

	5 Agents	*7 Agents*	*9 Agents*	*10 Agents*	*16 Agents*
Justification	88.50	88.86	88.00	88.35	87.28
Committee	88.36	87.90	88.00	88.14	85.71
Individual	78.78	76.43	72.07	68.36	67.07

Table 1. Accuracy comparison between the justification and the committee approaches in the unbiased scenarios.

	5 Agents	*7 Agents*	*9 Agents*	*10 Agents*	*16 Agents*
Justification	88.78	88.78	88.36	88.86	87.93
Committee	84.36	85.86	86.50	87.00	85.61
Individual	70.85	68.78	66.07	65.5	63.86

Table 2. Accuracy comparison between the justification and the committee approaches in the biased scenarios.

comparison purposes, we have tested the accuracy obtained if we have a system containing only one agent (having the whole training set), and we have obtained an accuracy equal to 88.2%.

5.2. Biased Scenarios

In the biased scenarios, the training set is also divided in as many subsets as agents in the system, but this time, each subset in not chosen randomly. We have forced some of the agents in the system to have more cases of some classes and to have less cases of some other classes. There can be also agents that don't have cases of some of the classes, or even agents that have only cases of a single class. This bias diminishes individual accuracy (Ontañón & Plaza, 2002).

Table 2 shows the classification accuracies obtained by several groups of agents using the JEC policy and the committee collaboration policy in the biased scenarios. As we have said, the individual accuracies when the agents have biased case bases are weaker than in the unbiased scenarios. This weakening in the individual results is also reflected in the Committee accuracy. This can be seen in Table 2, where the accuracy achieved by the Committee is clearly lower than the accuracy obtained in the unbiased scenarios shown in Table 1. However, comparing the results of Tables 1 and 2, we can see that the results obtained by the agents using the JEC policy are not affected by the bias of the case bases. Therefore, the increase in accuracy obtained with the JEC policy is greater in the biased scenarios. For instance, in the 5 agents scenario, the accuracy obtained with the JEC policy is 88.78% and the accuracy with the committee policy is only 84.36%, while the accuracy for the 10 agents scenario is 87.93% versus 85.62%. Again, we can see here how the confidence measures are independent of the case distribution among the agents, ensuring a robust aggregation of the individual predictions.

5.3. Redundancy Scenarios

In the redundancy scenarios, the training set is divided into subsets that are not disjoint, i.e. there are cases that are present in more than one subset. To create these disjoint subsets, we have duplicated 50% of the cases in the training set before distributing it among the agents. This means that if the original training set had 252 cases, the new training set has 378 cases (126 of them occur twice). We have said that the computation of the confidence values of the JERs is not sensitive to the distribution of cases among the agents, however the amount of redundancy in the system does affect it. These scenarios have been created to test the sensitivity of the computation of the confidence values of the JERs to redundancy.

Table 3 shows the accuracy obtained by both the JEC policy and the committee policy (the individual accuracy is also shown for comparison purposes). The first thing we see is that the JEC policy again outperforms the committee policy. This difference is very clear in the scenarios with many agents (9, 10 and 16). For instance, in the 16 agents scenario, the JEC policy obtained an accuracy of 88.28% and the committee policy just 87.71%. However, in these scenarios, the two policies are closer than in the previous two. For instance, in the 5 and 7 agent scenario, it is not so clear which policy works better: 89.64% for the JEC policy versus 89.78% in the 5 agents scenario, and 89.57% versus 89.50% in the 7 agents scenario.

If we look at the individual results in Table 3, we notice that they are much better than the individual results without redundancy: this is due to an increase of size in the individual case bases. This greater individual accuracy also enables the committee and JEC policies to obtain higher accuracies.

Notice that in the redundancy scenario there is overlapping among the agent's case bases. The examina-

	5 Agents	*7 Agents*	*9 Agents*	*10 Agents*	*16 Agents*
Justification	89.64	89,57	89.64	88.57	88.28
Committee	89.78	89.50	88.75	88.42	87.71
Individual	83.00	81.00	78.21	76.64	70.92

Table 3. Accuracy comparison between the justification and the committee approaches in the redundancy scenarios.

tion process is sensitive to the degree of overlapping because there are cases that (as present in several case bases) may be counted more than once for computing confidence estimates. However, as the results show, the JEC policy is robust enough to keep performance equal or better than the committee policy.

6. Conclusions and Future Work

We have presented a method to aggregate predictions coming from several agents into a single prediction using symbolic justifications. Allowing the agents to give a justification of their individual results is crucial in multiagent systems since in an environment where one's conclusions may depend on knowledge provided by third parties, justifications of these conclusions become of prime importance (van Harmelen, 2002).

In the experiments section, we have shown that both Committee and JEC policies obtain higher accuracies than the individual accuracy. However, the JEC policy is more robust than the Committee policy. Committee works well when the individual agents can provide good individual predictions. However, as our experiments show, when the individual accuracy drops, the accuracy of the Committee also drops. The JEC policy is more robust because the accuracy decreases much less (as can be seen in all the 16 agents scenarios). Moreover, when the distribution of the data among agents is not uniform and the individual agents have a skewed view of the data, the JEC policy is still able to keep the accuracy at the same level. More favorable scenarios for the Committee policy are the redundancy scenarios, where the individual agents have higher individual accuracies —although the JEC policy is still equal or better than the Committee.

We have only tested our method using LID as the classification algorithm, but in fact, any method that can provide a symbolic justification of the prediction can be used. As future work, we plan to evaluate the approach with decision trees, from which a symbolic justification can be constructed.

We have seen that the confidence estimation is independent of both the number of agents in the system and the distribution of cases among them. However, it is dependent on the degree of redundancy among the agent's case bases. In the future, we plan to use a new confidence estimation system that takes into account the redundancy level among the agent's case bases.

References

Armengol, E., & Plaza, E. (2001). Lazy induction of descriptions for relational case-based learning. *ECML 2001* (pp. 13–24).

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*, 123–140.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proc. 13th ICML* (pp. 148–146). Morgan Kaufmann.

Gama, J. (1998). Local cascade generalization. *Proc. 15th ICML* (pp. 206–214). Morgan Kaufmann, San Francisco, CA.

Hansen, L. K., & Salamon, P. (1990). Neural networks ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 993–1001.

López de Mántaras, R. (1991). A distance-based attribute selection measure for decision tree induction. *Machine Learning*, *6*, 81–92.

Ontañón, S., & Plaza, E. (2001). Learning when to collaborate among learning agents. *12th European Conference on Machine Learning* (pp. 394–405).

Ontañón, S., & Plaza, E. (2002). A bartering aproach to improve multiagent learning. *AAMAS 2002* (pp. 386–393).

Ontañón, S., & Plaza, E. (2003). *Justification-based multiagent learning* (Technical Report). Artificial Intelligence Research Institute (IIIA). TR-2003-07.

Perrone, M. P., & Cooper, L. N. (1993). When networks disagree: Ensemble methods for hydrid neural networks. In *Artificial neural networks for speech and vision*. Chapman-Hall.

van Harmelen, F. (2002). How the semantic web will change KR. *The Knowledge Engineering Review*, *17*, 93–96.