Apprentissage Multi Agent à Memoire Bornee

Gauvain Bourgne, Henry Soldano, Amal El Fallah Seghrouchni

Apprentissage Multi Agent à Mémoire Bornée

Gauvain Bourgne[1], Henry Soldano[2], Amal El Fallah Seghrouchni[3]

[1] CRIL, UMR-CNRS 8188, Université d'Artois, 62300 Lens, France
[2] LIPN, UMR-CNRS 7030, Univ. Paris-Nord, 93430 Villetaneuse, France
[3] LIP6, Univ. P. et M. Curie 104, Av. du Pr. Kennedy, 75016 Paris, France

Résumé : Nous abordons ici l'apprentissage supervisé en ligne collaboratif dans une société d'agents. La démarche adoptée est celle du maintien collectif d'une notion de consistance, ici correspondant au maintien, par révision de l'hypothèse courante, d'une hypothèse d'erreur empirique nulle. L'hypothèse prend la forme d'une formule de taille réduite et la révision repose sur les exemples mémorisés. Lors de précédents travaux, dans le cadre du projet SMILE, tous les exemples rencontrés par un agent, plus ceux transmis par d'autres agents, étaient mémorisés. Dans le travail présenté ici, chaque agent a une mémoire bornée, limitant ainsi le nombre d'exemples maintenus dans la mémoire de chaque agent. Nous proposons une adaptation du mécanisme de révision collective de SMILE prenant en compte cette restriction. Plusieurs variantes de ce mécanisme, se différenciant en particulier selon la méthode utilisée par les agents pour gérer leur mémoire, sont explorées expérimentalement. Nous observons alors dans quelle mesure ces restrictions en mémoire peuvent être dépassées résultant parfois de manière surprenante en une erreur en test plus faible que sans ces restrictions.
Mots-clés : Apprentissage collectif, Apprentissage collaboratif, Apprentissage Multi Agents, Apprentissage incrémental.

1. Introduction

This article deals with the problem of collaborative learning (Panait & Luke, 2005) in a multi-agent system (MAS) when agents have a limited storage capacity and therefore cannot keep all the examples they receive from the environment or other agents. Memory (viewed as an internal resource of an agent) may be bounded because of one or a combination of several reasons : 1) to reduce the cost of saving or accessing examples 2) to reduce the complexity and hence the cost of inference/learning 3) to allow the agents deployment on small and mobile devices (smartphones, tablets, etc. offer small memories for computation). In MAS literature, several capabilities of agents have been bounded. Resource-bounded agent refers to an agent with limited access to resources to compute plans or to make decisions. These resources are mainly external and shared with other agents. In addition, these shared resources may vary at runtime (Bratman *et al.*, 1988). In our approach, agent memory is internal and stable over the time. An agent has a full control on his memory and the way it is managed.

In this paper a model of bounded memory is proposed. Several experiments are conducted to study the compromise between efficient learning and the volume of examples the agent has to keep in his memory. The main question tackled in this paper is to construct a computational rational-learning theory for cognitive agents subject to realistic memory constraints. More precisely, we are concerned with the extension of *online* concept learning from examples, a simple model of supervised learning that outputs a hypothesis that is supposed to cover positive examples and to reject negative examples of some target concept, to a collaborative setting.

At the agent level this a simple case of single agent learning, in which the agent observes a stream of examples of some target concept all along his life. Learning is then *online* as the agent sequentially receives examples, predicts and observes their label, and considers each misclassified example as a *counter-example* of the current theory representing the target concept. Furthermore, we consider that the agent is an *incremental* learner, i.e. when new examples occur, the agent *revises* the theory, rather than building from scratch a new theory from past and new examples. Among incremental learners, *no memory* learners only stores the current theory, *partial memory* learners also memorize part of the previous examples, and *full memory* memorize all the previous examples together with the current theory (Maloof & Michalski, 2004). Full memory and partial memory learners maintain the current solution as simple as possible, and the available memory in this case is mostly used for storing the examples.

Various strategies for selecting relevant examples in partial memory learners have been investigated, both concerning logics based learning algorithms as rule learning algorithms (Maloof & Michalski, 2004), and numerical methods as Support Vector Machines (Domeniconi & Gunopulos, 2001), among which i) selecting the examples within a certain time window, useful when the target concept is suspected to change over time (drift concept), and ii) selecting counter-examples. What we investigate here also needs some selection scheme, as when the memory limit is exceeded, storing a new example means that some previous example has to be forgotten (Laskov *et al.*, 2006). We first propose hereunder to investigate the impact of limiting the available memory in single agent learning, in particular when prioritizing counter-examples in the selection process.

At the multi agent level, collaborative concept learning strategies have been recently investigated, with the purpose of merging hypotheses built, in a batch way, by several agents (Ontañón & Plaza, 2010), or of having in turn each agent revising the current theory. We consider here the latter approach, resulting in the SMILE implementation (Bourgne *et al.*, 2007). In SMILE, autonomous agents are organized in a fully connected MAS, and each agent stores examples received from the environment or from other agents. Each agent also stores and shares with the other agents, a current *theory* common to all the agents in the MAS. When an agent receives from the environment some counter-example, denoted as *internal*, it has first to revise the current theory in order to keep it *consistent* with its own example memory. However, as it has also to keep the hypothesis consistent with the whole information in the MAS, a set of interactions with the other agents is necessary. During these interactions, the revising agent plays the role of the *learner*, proposing a revision of the current theory while the other agents acting as *critics* either accept the revision or return an example contradicting the revised theory and denoted, in the learner agent view, as an *external* counter-example. Agents can in turn be learners or critics, none of them being kept to a specific role. Further work addressed the question of constraints on the communications (Bourgne *et al.*, 2009) or the effect of broadcasting the revised hypothesis to the critics (Bourgne *et al.*, 2010).

The revision mechanism at the core of the collaborative multi agent learning protocols implemented in SMILE is a full memory learner. We then investigates the effect of limiting the storage capacity of these agents on collaborative concept learning. As we will see, the simple priority scheme proposed at the agent level has to be adapted.

2. Single agent and multi agent incremental learning

2.1. Learning task

We consider here concept learning in which a learner tries to build a representation (or hypothesis) for recognizing positive examples of a given concept. It can be easily extended to classification (learning several classes instead of a single concept), but in order to simplify the analysis we shall remain in this framework.

In this context a hypothesis H is a monotone DNF, i.e. a disjunction of terms h, each represented as a conjunction of positive literals from a set of atoms A. An example is an interpretation together with a label $+$ or $-$ indicating whether it belongs or not to the target concept. A hypothesis H *covers* an example e whenever e satisfies (is a model of) H [1]. Given a set of positive and negative examples $E = E^+ \cup E^-$, a hypothesis is *complete* when it covers all the positive examples in E^+, and is *coherent* when it covers no negative examples in E^-. To learn DNF, negative literals are represented by additional atoms, like $not - a$. A target formula as for instance f =$(a \wedge b) \vee (b \wedge \neg c)$ would thus be represented as $(a \wedge b) \vee (b \wedge not - c)$. The positive example $\{not - a, b, not - c\}$ is then a model of f.

After learning on a set of training examples (with labels), the hypothesis is to be used on an unlabeled example. If the hypothesis covers such an example, it will be predicted as being positive, otherwise, it will be predicted as negative. In our experiments, to evaluate the quality of a hypothesis, we shall thus measure its *accuracy*, that is, the percentage of correctly classified examples in a test set.

2.2. Incremental learning

Given a current hypothesis H, complete and correct with respect to a memory state $E = E^+ \cup E^-$ filled with the previous examples, and a new positive or negative example e, the revision mechanism produces a new revised hypothesis complete and correct with respect to the new memory state including e. We describe below our single agent revision mechanism, inspired from a previous work on incremental learning (Henniche, 1994).

In this process, each term is the *lgg* (least general generalization) of a subset of positives instances $\{e_1, ..., e_n\}$ (Fürnkranz, 2002), that is the most specific term covering $\{e_1, ..., e_n\}$. The *lgg* operator is defined by considering examples as terms, so we denote as $lgg(e)$ the most specific term that covers e, and as $lgg(h, e)$ the most specific term which is more general than h and that covers e. Restricting the term to *lgg* is the basis of a lot of Bottom-Up learning algorithms (for instance Fürnkranz (2002)).

---

1. we identify e to the term made of its positive atoms. So e is a model of H whenever there is a term t in H *more general than* e, i.e. such that t is included in e.

The revision mechanism, originally a full memory mechanism, depends of the ongoing hypothesis H, the ongoing examples E^+ and E^-, and the new example e. There are three possible cases :

- e is positive and H covers e, or e is negative and H does not cover e. No revision is needed, H is already complete and coherent with $E \cup \{e\}$.
- e is positive and H does not cover e : e is denoted as a *positive counterexample* of H. Then we seek to generalize in turn the terms h of H. As soon as a correct generalization $h' = lgg(h, e)$ is found, h' replaces h in H. If there is a term that is less general that h', it is discarded. If no generalization is correct (meaning here coherent), $H \cup lgg(e)$ replaces H.
- e is negative and H covers e : e is denoted as a *negative counterexample* of H. Each term h covering e is then discarded from H and replaced by a set of terms $\{h'_1, ..., h'_n\}$ that is, as a whole, coherent with $E^- \cup \{e\}$ and that covers the examples of E^+ uncovered by $H - \{h\}$.

Terms of the revised hypothesis H that are less general than others are discarded from H.

Note that this mechanism tends to both make a minimal revision of the current hypothesis and to minimize the number of terms in the hypothesis. The new example is tagged as a counter-example. If it is a negative counter-example, all positive examples explicitly used during the revision to generate new terms are also tagged as counter-examples in the agent memory.

2.3. Collaborative Learning

SMILE adopts a learner/critic approach in which an agent revising the hypothesis (the learner) proposes it to all the other agents, acting as critic, to obtain counter examples allowing it to improve the hypothesis if needed.

2.3.1. SMILE mechanism

In SMILE, a MAS with n agents, denoted n-MAS, is represented as a set of agents a_1, ..., a_n. Each agent a_i has a hypothesis H_i and an example memory E_i. Learning is seen as the maintenance of some *consistency* property $Cons(H, E)$ between a hypothesis H and a set of examples E, meaning here that the hypothesis is coherent and complete w.r.t. E.

Definition 1

An agent a_i is a-consistent *iff $Cons(H_i, E_i)$ is true. An agent a_i is* mas-consistent *iff $Cons(H_i, E)$ is true, where $E = \cup_{j \in \{1,..,n\}} E_j$ is the information stored in the n-MAS. A n-MAS is* consistent *iff all its agents r_i are mas-consistent.*

Consistency of the agents is *additive*, meaning that whenever $Cons(H_i, E_1)$ and $Cons(H_i, E_2)$ hold, $Cons(H_i, E_1 \cup E_2)$ also holds.

M denotes an *internal revision* mechanism that is applied whenever the agent a_i receives a counter-example e, turning E_i into $E'_i = E_i \cup \{e\}$ such that $Cons(H_i, E'_i)$ is false. It changes H_i into a new hypothesis $H'_i = M(H_i)$. This mechanism should be *a-consistent*, meaning that is should ensure that $M(H_i)$ is consistent with E'_i, therefore restoring the a-consistency of the agent. In practice, M will be the revision process described previously. In the same way, the *mas-consistency* of a *revision* mechanism M_s requires that the agent stays consistent with the whole information stored in the MAS. Finally M_s is *strongly mas-consistent iff* when M_s is applied by an agent, the whole MAS is made consistent.

The revision mechanism M_s proposed in SMILE is then constituted of reiterated applications by the *learner* agent a_i of its *internal* a-consistent revision mechanism M, followed by some interactions between a_i and the other agents, until a_i regains its mas-consistency. The mechanism is triggered by an agent a_i that, upon receipt of a *counter example* e, revises H_i to H'_i. An interaction $I(a_i, a_j)$ between the *learner* agent a_i and another agent a_j, acting as *critic* is as follows :

1. agent a_i sends the revision H'_i to a_j.
2. agent A_j checks the revision H'_i. If this hypothesis is consistent with its own examples, a_j sends to a_i an *acceptance* of H'_i, else it sends a contradictory example e' : $Cons(H'_i, e')$ is false.

An iteration of M_s is then composed of an internal revision performed by the *learner* agent a_i, followed by a sequence of interactions $I(a_i, a_j)$. If a counter example e' is transmitted to a_i, this triggers a new iteration, starting with a new revision of the learner to restablish its consistency. Otherwise, when all the critics have sent an acceptance of the proposed hypothesis H'_i, a_i has restored its mas-consistency. It then notifies the other agents, who *adopt* the new hypothesis $H' = H'_i$. This ensures that, at the end of the revision process, all the agents share the same hypothesis H'.

In Bourgne *et al.* (2007), the revision mechanism M_s described above was proved as strongly mas-consistent when $Cons$ is additive. The experiments showed in particular an unexpected accuracy increase, regarding hard Boolean problems, when comparing multi agent learning to single agent learning from the same m examples.

3. Memory bounded single agent learning

3.1. Managing the bounded memory

We shall consider that agents always keep a small, fixed amount of memory for storing their hypotheses. The memory bound will be represented as the number of training examples that an agent can keep in memory. A memory-bounded learning agent a_i with memory bound k with thus be represented as a pair (H_i, E_i) where H_i is the current hypothesis and where the example memory E_i is always such that $|E_i| \leq k$. As a result, once the example memory is full, an agent cannot store new examples without first forgetting other ones.

When receiving a new example e, an agent first uses the revision mechanism described above to revise, if needed, its current hypothesis, then he applies some memory management process to decide which examples should be kept. If the memory is not yet saturated ($|E_i| < k$), the agent can just store the new example e, but otherwise, it must either discard this example, or choose to forget another example from its current memory in order to store e. We propose two strategies from memory management. In the *basic* strategy make the agent forget old examples to store new ones. The example memory then behaves as a bounded-size *First In First out* list.

As a classical selection strategy in partial memory learners consists in memorizing only the counter examples, we also propose a *prioritized* strategy : counter-examples have a higher priority than other examples, and, within the same priority level, older examples are forgotten to make room to new ones. An agent will thus store all examples until it has received k examples, then it will store new examples by forgetting the oldest low-priority example until the memory is saturated with high priority examples. From that point onward, the agent will only store new high priority examples, forgetting his oldest example to do so (low-priority example being directly discarded after revision).

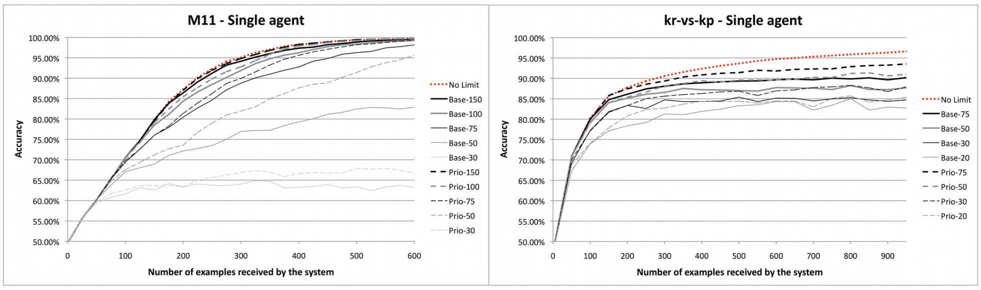

FIGURE 1: Accuracy of a single agent with different memory bounds k and either basic (Base-k, plain lines) or prioritized strategy (Prio-k, dotted lines).

3.2. Experiments

We have performed experiments on two concept learning problems with various memory bounds. In each case we tested both the basic strategy of keeping the k most recent examples (Base or B) and the prioritized strategy where counter examples are preferred (Prio or P). 50 trials were run for each strategy, using the same 50 random sequences of training and testing examples. The two problems investigated, taken from Bourgne *et al.* (2007), are a difficult boolean problem, the Multiplexer-11 (M11), and the kr-vs-kp problem taken from the UCI ML database.

Fig. 1 displays the learning curves for increasing memory bounds on the M11 and kr-vs-kp problems. As the memory size k increases, the learning gets closer and closer to the learning curve of an agent without memory limitations. The figure also confirms that giving priority to the storage of counter-examples (dashed lines) does increases the accuracy with respect to the basic strategy. Finally, consider that in an online memory bounded setting, a non incremental learner would learn from scratch, using only the memorized examples, whatever is the number of previous examples. In the M11 case, after $k = 100$ examples, we observe Figure 1 an accuracy about 70% whereas the agent reaches more than 96% after 400 examples with the same memory size [2].

2. Here we estimate the accuracy of a non incremental learner by observing an incremental learner starting from the empty hypothesis.

4. Memory Bounded Multi Agent Learning

4.1. Adapting the selection strategies to multi agent learning

Bourgne *et al.* (2010) introduces *forgetness* to ensure minimal redundancy in SMILE : at the end of the revision mechanism, the learner agent forgets all the external counter-examples he received from other agents. In the bounded memory setting this seems a straightforward way to ensure that there will always be as much as possible distinct examples in the whole memory of the multi agent system. However, there is no guarantee that a memory bounded agent will have enough memory to temporarily memorize all the necessary counter-examples before being allowed to forget them. For instance, in the M11 problem, the learner has to exchange up to 72 examples before restoring the mas-consistency of his hypothesis. Still, as examples received from other agents are redundant, the agents differentiate them in their memory by tagging them as *external*, while examples received from the environment are tagged as *internal*.

To perform learning in the context of a system of memory-bounded agents, the basic idea is to apply the collaborative learning presented before, where each agent, when receiving a new example, applies his memory management process *after* revising his hypothesis (if needed), and *before* sending any message. The problem is that if the agent encounters during the revision more external examples than he can store, he will forgot some of them, the current revised hypothesis may become inconsistent with these forgotten examples, and termination of the revision process is not guaranteed anymore. To restore termination, we proceeds as follows. Whenever a critic send a counter example during a revision, he will tag the example as "Sent-1". On subsequent critique, the critic agent will check the tags of the counter-example it intends to send. If it has the "Sent-n" tag, it will replace this tag by "Sent-$n + 1$" before sending it. To ensure termination, we have to bound the number of time the same example can be sent. We shall thus decide on a *repeat bound* b. When $n + 1 = b$, instead of tagging the example with "Sent-b", the agent will put the tag "Ignore". All examples tagged with "Ignore" are not considered during the critique. They will not be sent again during this same revision. This ensures that the global revision will terminate (with at most $b * n_{allExt}$ counter-examples sent, where n_{allExt} is the cardinality of the union of the example memory of all critics).

We call such a mechanism RB-SMILE-Rb. Mas-consistency cannot be guaranteed anymore, but we can still ensure a weaker property :

Property 1
At the end of a revision with RB-SMILE-Rb, the hypothesis of the learner is consistent with all examples in the system, except those tagged with "Ignore".

Proof This follows from the definition of the mechanism. A critic will only send accept if all its examples that are not tagged with "Ignore" are consistent with the hypothesis, and the revision will only stops once the learner agent has received acceptation of the same hypothesis from all the critic agents. □

At the end of a global revision, all critics remove the "Sent-n" and "Ignore" tags from all their examples.

Now the question is : how should we adapt the memory management strategies to the multi agent case ? The Basic strategy can be directly applied, but regarding the Priority strategy, some more care is needed. Indeed, all examples exchanged during a revision are counter-examples, and if we prioritize all counter-examples, we are favoring redundancy in the system. We would like to both favor *importance* of the stored examples, as in centralized case, and *diversity* of the stored examples, but minimizing redundancy. We propose here to state as high priority examples, for a given agent, his *internal counter-examples*. As a result, if his memory becomes saturated with internal counter-examples, an agent will not store any external counter-examples, just using them for revising his hypothesis before discarding them. Note that diversity does matter, as experiments, not discussed here, giving all counter examples high priority (whether internal or internal) actually resulted in a worst accuracy than the basic strategy.

We shall now present experiments on using collaborative learning with memory-bounded agents, and discuss the influence of the different parameters (repeat bound, memory bound, number of agents).

4.2. Experimental study

The problems, taken from Bourgne *et al.* (2007), are the M11 problem, two parity problems defined on 3 (5) bits with 25 (5) irrelevant attributes (Xor3-25 and Xor5-5) and three problems taken from the UCI ML database for which the full memory algorithm finds solutions with accuracy higher than 0.95 : tic-tac-toe, kr-vs-kp and voteMp. The two first are close to boolean problems, while the latter has nominal and numerical attributes.

During each trial, corresponding to a random sequence of examples taken in the learning set, the first example is sent to the first agent, then, once revision is finished, the second example is sent to the second agent, and so on, each agent getting in turn one example from the system.

We tested different memory bounds (k=n_M, with n_M varying from 15 to 75 depending on the problem) or repeat bounds (Rb, with b begin 1, 2, 5, or 10), as well as different system size (5.10. 20 agents for the M11, 10 agents only for the other problems). We focus on two main measures : accuracy and global execution time.

Table 1 gives the accuracy and time for different memory bounds k for the 6 problems, as well as the result with no memory limitation (last column). For each configuration, results are given for both basic and prioritized strategy (priority to internal counter examples) with a repeat bound of either 1 (R1, each counter-example is only given once) or 10 (R10, a given counter-example can be sent as much as 10 times if needed). We will now discuss those results starting with boolean problems.

Influence of the number of agents. Fig.2 compares the accuracy results of different number of agents with or without priority. In both case, we can see that with tight bounds, increasing the number of agents is really helpful in avoiding too much loss of accuracy. When the memory bound is looser, collaborative learning allow to recover the original accuracy really quickly, and adding more agents do not have much more benefit beyond a certain point. Concerning efficiency, as in SMILE, the costs are increasing with the number of agents. This increases is sharper for tight bounds. For basic strategy with R1, we need respectively 3.6s, 14.6s and 45s for 5, 10 and 20 agents when k=30. When k=50, those times become 0.9s, 1.8s and 3,8s, whereas without limit (SMILE), we need respectively 1.0s, 2.0s and 3.9s for 5, 10 and 20 agents. Tight bounds clearly increase the number of computations and communications, as useful examples are quickly forgotten generating many revisions.

Influence of the repeat bound. Fig.3 compares the accuracy results of different limit bounds for different repeat bounds (R1, R2, R5, R10) for the M11 with 10 agents and the priority strategy, with a tight and loose memory bound. It is clear that this parameter has almost no effect when the memory bound is loose enough, as the critics do not need to send several time the same counter-example When the bound is tight, however, a clear difference appear. In such a restricted setting, learner does forget the first counter-examples during the revision, and giving them more opportunities to adjust the hypothesis significantly increases the accuracy (e.g. from accuracy of 82.8% with R1 at 200 examples to accuracy of 94.3% with R10). This also explains why the lear-

TABLE 1: Accuracy results of 10 agents learning

Problem		Base R1	Base R10	Prio R1	Prio R10	No bound
M11	k=20	66.1	68.9	62.6	T.O.	
(200 ex.)		*18s*	*80s*	*35s*		
	k=30	80.2	89.7	82.8	94.3	93.9
		15s	*13s*	*20s*	*21s*	*2.0s*
	k=50	95.1	94.9	95.6	96.0	
		1.8s	*1.8s*	*1.7s*	*1.7s*	
Xor3-25	k=15	69.5	78.2	71.6	95.7	
(100 ex.)		*7.1s*	*9.5s*	*11s*	*11s*	
	k=20	88.2	90.5	94.1	97.1	80.5
		2.9s	*2.8s*	*3.7s*	*3.2s*	*4.3s*
	k=25	89.7	91.2	94.2	95.6	
		2.7s	*2.5s*	*2.8s*	*2.4s*	
	k=30	90.0	90.7	91.7	92.9	
		2.9s	*3.0s*	*2.8s*	*3.0s*	
Xor5-5	k=40	59.4	64.6	59.4	72.7	
(100 ex.)		*46s*	*193s*	*86s*	*206s*	
	k=50	85.2	85.7	85.8	90.2	88.3
		8.2s	*8.6s*	*15s*	*15s*	*4.8s*
	k=75	89.6	89.7	88.2	88.3	
		4.4s	*4.4s*	*4.6s*	*4.7s*	
VoteMp	k=15	94.6	94.5	93.6	94.4	
(391 ex.)		*1.4s*	*2.0s*	*2.7s*	*12s*	
	k=20	93.9	94.0	94.1	94.5	
		2.8s	*3.3s*	*4.2s*	*19s*	93.4
	k=30	93.1	92.6	94.1	94.9	*9.1s*
		4.7s	*4.9s*	*7.3s*	*31s*	
	k=50	92.6	93.1	93.3	93.0	
		5.4s	*5.3s*	*8.1s*	*12s*	
kr-vs-kp	k=20	90.7	90.8	88.5	90.2	
(958 ex.)		*27s*	*36s*	*50s*	*224s*	
	k=30	93.8	93.5	92.0	92.9	
		29s	*43s*	*67s*	*292s*	97.5
	k=50	95.1	96.1	95.3	95.7	*32s*
		39s	*45s*	*64s*	*215s*	
	k=75	96.9	97.0	97.5	97.2	
		34s	*34s*	*49s*	*85s*	
tictactoe	k=20	74.5	76.2	76.9	82.7	
(100 ex.)		*3.2s*	*2.2s*	*6.7s*	*6.8s*	
	k=30	79.5	79.4	83.6	82.3	78.5
		0.78s	*0.78s*	*0.74s*	*0.77s*	*0.75s*
	k=50	78.7	78.3	80.2	80.6	
		0.75s	*0.75s*	*0.76s*	*0.77s*	

ning takes more time with such tight bounds. The increases of the cost of those revision is well compensated by the fact that as the hypothesis gets more accurate, less revisions are needed, so overall, increasing the repeat bound does not really increase the execution time (see boolean problems on Table1 with R1 and R10), and sometimes even decreases it.

Influence of the memory bound. Fig.4 compares the accuracy results of va-

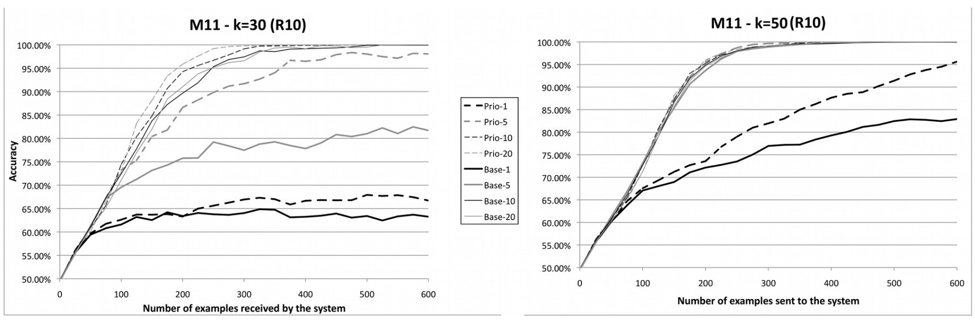

FIGURE 2: Accuracy results for the M11 problem with a repeat bound of 10 and n =1, 5,10 or 20 agents, with basic strategy (Base-n, plain lines) or prioritized strategy (Prio-nm dotted lines). Results are given for a memory of k=30 (left) or 50 (right).

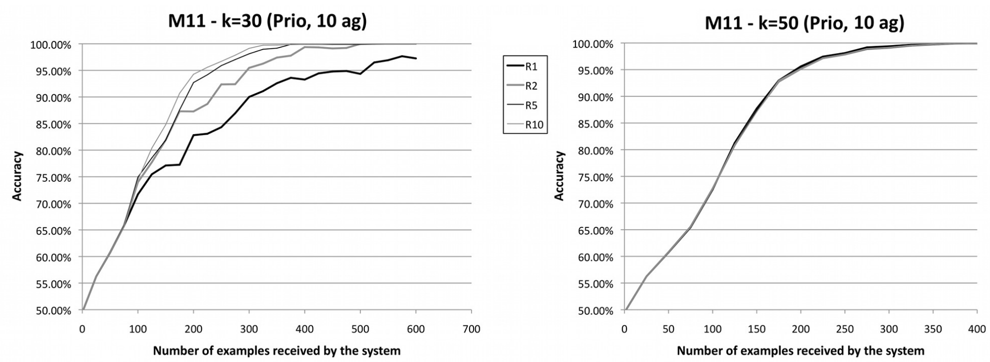

FIGURE 3: Accuracy results for the M11 problem with a repeat bound R of 1, 2, 5,10 or 20 for 10 agents, with prioritized strategy. Results are given for a memory of k=30 (left) or 50 (right).

rious memory bounds for two problems, with 10 agents, the priority strategy, and a repeat bound of 1 (R1), As said before, when the memory limit is low, learning becomes difficult, and the accuracy decreases, while a very loose memory limit give similar results as unbounded memories. However, in the middle zone, we see here a very interesting phenomenon, as accuracy is in fact *increased* by the memory bound.

A tight memory bounds enforces more revisions in order to satisfy all critics, and when this does terminate without ignoring too many examples , we benefit from this more intensive exploration of the hypothesis space. This is the same phenomenon that was argue to explain the better performance of collaborative learning with respect to individual learning in SMILE (Bourgne *et al.*, 2010). Moreover, since each agent does not have many examples in memory, critique and learning computational cost are lower than in the normal case (no limit), and, for example, Xor3-25 with k=20 gives a better accuracy than classical SMILE, with less execution time (see Table 1). Likewise, tic-tac-toe with k=30 has a better accuracy than its unbounded counterpart, for almost the same execution time.

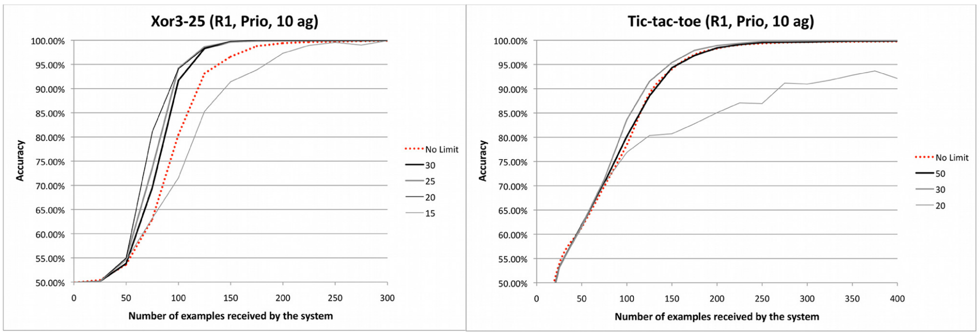

FIGURE 4: Accuracy results for 2 (pseudo) boolean problems with various memory bounds for 10 agents, with prioritized strategy and a repeat bound of 1.

Influence of the priority. Prioritizing the internal counter examples improves the accuracy on boolean problems (see Table 1), except when the memory bound is too tight to allow convergence. When the memory bound is low (30 for M11, 15 for Xor3-25, 20 for tic-tac-toe, 40-50 for Xor5-5), execution time is higher than the basic strategy, but for looser bounds, it become equivalent.

4.2.1. Database problems

Fig.5 gives the accuracy results for two UCI database problems, with increasing memory bounds, Here, we do not have the accuracy improvement observed in boolean problems. For kr-vs-kp, the accuracy gets better as the bound gets looser, up to the accuracy of unbounded case. For voteMp, however, the bound limit hardly seem to affect the accuracy. With k=15, we quickly reach 94% accuracy but cannot get past it afterwards. Allowing a high repeat bounds with priority in both problem really increases the cost, with no benefit on accuracy for voteMp. Since this strategy increase the cost of individual revision, it becomes costly when the learning does not converge to a perfect solution, as the number of revision is not reduced when compared to other strategies. Using priority on counter examples on these problems means that we might end up giving more importance to some noisy examples, and this strategy does not performs better than the basic one. Basic strategy with single repeat is in these cases the less costly, and it takes less time than the unbounded strategy in most case.

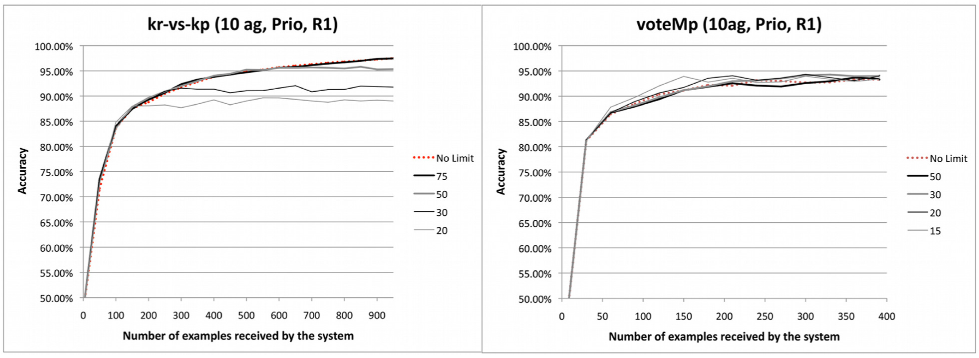

FIGURE 5: Accuracy results for 2 UCI database learning problems with various memory bounds for 10 agents, with prioritized strategy and a repeat bound of 1.

5. Conclusion

In this paper we have investigated online learning in which a multi-agent system (MAS) performs revisions of its current theory by maintaining some notion of MAS consistency. In realistic context, the constraint of bounded memory of the agents is meaningful and raises several fundamental questions related to the issue of collaborative learning either from single agent or MAS perspectives. In this work several experiments have been successfully conducted and have provided interesting (and sometimes non-expected) answers to some of these questions. For instance, what is the impact of using collaborative learning with memory bounded agents, and what is the influence of the different parameters (repeat bound, memory bound, and number of agents) ? How the MAS overcomes the memory restrictions of its agents, still taking advantage of the examples distributed in the agent memories ? How to adapt collaborative learning to a memory-bounded context to overcome the bounded memory restriction, and even get better accuracy or efficiency in some cases. In future work, we plan to extend this framework and to add another realistic constraints related to the communication between agents and the topology of the underlying network.

Références

BOURGNE G., ELFALLAH-SEGHROUCHNI A. & SOLDANO H. (2007). Smile : Sound multi-agent incremental learning ;-). In *AAMAS'07*, p. 164–171 : IFAAMAS.

BOURGNE G., ELFALLAH-SEGHROUCHNI A. & SOLDANO H. (2009). Learning in a fixed or evolving network of agents. In *ACM-IAT'09* : IEEE.

BOURGNE G., SOLDANO H. & EL FALLAH SEGHROUCHNI A. (2010). Learning better together. In H. COELHO, R. STUDER & M. WOOLDRIDGE, Eds., *ECAI*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, p. 85–90 : IOS Press.

BRATMAN M. E., ISRAEL D. J. & POLLACK M. E. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence*, **4**(3), 349–355.

DOMENICONI C. & GUNOPULOS D. (2001). Incremental support vector machine construction. In N. CERCONE, T. Y. LIN & X. WU, Eds., *ICDM*, p. 589–592 : IEEE Computer Society.

FÜRNKRANZ J. (2002). A pathology of bottom-up hill-climbing in inductive rule learning. In *ALT*, volume 2533 of *LNCS*, p. 263–277 : Springer.

HENNICHE M. (1994). Mgi : an incremental bottom-up algorithm. In *IEEE Aust. and New Zealand Conference on Intelligent Information Systems*, p. 347–351.

LASKOV P., GEHL C., KRÜGER S. & MÜLLER K.-R. (2006). Incremental support vector learning : Analysis, implementation and applications. *J. Mach. Learn. Res.*, **7**, 1909–1936.

MALOOF M. A. & MICHALSKI R. S. (2004). Incremental learning with partial instance memory. *Artif. Intell.*, **154**(1-2), 95–126.

ONTAÑÓN S. & PLAZA E. (2010). Multiagent inductive learning : an argumentation-based approach. In J. FÜRNKRANZ & T. JOACHIMS, Eds., *ICML '10 : Proceedings of the 27th Annual International Conference on Machine Learning*, p. 839 – 846 : Omnipress.

PANAIT L. & LUKE S. (2005). Cooperative multi-agent learning : The state of the art. *Autonomous Agents and Multi-Agent Systems*, **11**(3), 387–434.