Image Retrieval over Networks : Active Learning using Ant Algorithm

David Picard*, Matthieu Cord, and Arnaud Revel

***Abstract*—In this article, we present a framework for distributed content based image retrieval with online learning based on ant-like mobile agents. Mobile agents crawl the network to find images matching a given example query. The images retrieved are shown to the user who labels them, following the classical relevant feedback scheme. The labels are used both to improve the similarity measure used for the retrieval and to learn paths leading to sites containing relevant images. The relevant paths are learned in an ethologically inspired way. We made experiments on the trecvid 2005 keyframe dataset showing that learning both the similarity function and the localization of the relevant images leads to a significant improvement. We also present an extension with the re-use of learned paths for later sessions leading to a further improvement.**

I. Introduction.

With the generalization of networking devices and the expansion of interconnections, information has been spread over many sources. The Internet, p2p networks, professional or even personal intranets provide huge volumes of information. To tackle the search into these collections, search engines have been developed in order to find the best localizations of datas matching a query. They became efficient, user-friendly and very popular. They are even more relevant in dynamic networks like p2p, where the user is unable to crawl the network by hand. In that sense, the work on search engines is highly valuable for today's applications [1].

As far as data mining in multimedia documents (image, audio, video, ...) is concerned, web search engines usually give poor results. Most of them use the contextual web page, or the meta information attached to the multimedia objects. The text used for the indexing process is often far from the semantic meaning that the user usually attaches to the content of the document. Hence, the results of web search engines are far from expected regarding the semantics of the documents. Content Based Image Retrieval (CBIR) has been recently proposed in order to give an answer to this problem [2]. The main idea is to build a representation of the image based on its content, and then to find a relation between this representation and the semantic we associate to the image. Machine learning techniques have been successfully adapted to this framework. The best improvement was done with the introduction of relevance feedback [3], [4] into the process.

The processing cost introduced by these techniques makes them difficult to use with large amounts of images such as what we can find on the Internet. Moreover, classical CBIR tools are designed for a unique collection. In this paper, we adapt machine learning techniques such as active learning in order to deal with image retrieval distributed over a network. We propose to learn both the path leading to the collection containg the relevant images and the similarity between images. We introduce a scheme that efficiently implements this two-step learning combination by using an ant-like behavior algorithm. The resulting system will constitute a smart solution for distributed CBIR.

In the next section, bibliographical context is discussed, and our methodological and technological choices of mobile agents are motivated (Sect. III). Then, we detail the ant-like reinforcement learning algorithm used by the agents in section IV. The section V contains the description of our distributed interactive learning strategy. Finally, we present and discuss the experiments and results we obtained using our system on the trecvid2005 keyframe dataset[1].

II. Distributed content based retrieval

A. Content Based Image Retrieval

The main idea of content based image retrieval (CBIR) is to retrieve within large collections images matching a given query thanks to their visual content analysis. Visual features, such as color, texture or shape, are extracted from the images and indexed. These features can be global or extracted from regions or points of interest [5], [6] and then compiled into a index or signature. A basic way to perform retrieval consists in computing a similarity function for comparing the query index to one of the images in the collection [7]. Such systems give unsatisfying results due to the well known semantic gap [8]. To fill the gap, the similarity function may be updated on-line, using an interaction with the user called "relevance feedback". In this process, some images are proposed to the user for labeling. These labels (usually *relevant* or *irrelevant*) are used to update the similarity function.

In this context, machine learning techniques such as Bayes classification [9] or support vector machines [10] have been recently introduced to build the similarity function from a set of labeled samples. To boost this online learning, active learning strategies have been proposed: the goal is to select the examples to be labeled that will enhance the most the similarity function [11]. At the end of the interactive session, a ranking of all images given their similarity is shown to the user. Most of these techniques come from the text categorization community [12] and were adapted to image classification.

There are lots of CBIR systems, and an overview can be found in [13].

D. Picard is with ETIS CNRS UMR 8051, France. e-mail : picard@ensea.fr
M. Cord is with LIP6 UMPC, France. e-mail : matthieu.cord@lip6.fr
A Revel is with ENSEA and Centre Emotion CNRS UMR 7593, France. e-mail : revel@ensea.fr

[1]see http://www-nlpir.nist.gov/projects/tv2005/

B. CBIR in distributed collections

The major part of the CBIR computation is dedicated to the processing of all the feature vectors in order to produce the final ranking. Then, the fact that images are distributed over many sources should be more an advantage than a drawback since the processing of every image could be naturally paralleled. To our knowledge, there are only few researches on distributed image retrieval (despite being noted as further improvement to CBIR in [13]). In a context such as the Internet, the common idea is to keep up-to-date an index of the feature vectors. The indexes are stored on a few well identified machines hosting a CBIR search tool. These servers are entry points for the distributed CBIR system. The distributed CBIR system propagates the queries to the entry points and efficiently merges their results (see [14]). In peer-to-peer networks, it is not possible to identify entry points anymore. Instead, each peer must index its own images and queries must be propagated from one peer to another. In DISCOVIR [15], King proposes an algorithm for selecting links between peers based on the content of their shared images. The queries are more likely to be propagated to peers which are known to host similar images. With this method, they achieve to improve the retrieval and reduce the network load.

In the systems presented above, interaction and active learning are not taken into account. Thus, the question about efficiently performing active learning in a distributed context is still open.

C. Mobile agents

Among the strategies allowing to perform the needed parallelization of feature vector processing, we have chosen to use mobile agents. A mobile agent is an autonomous computer software with the ability to migrate from one computer to another and to continue its execution there. While mobile agents have motivated many researches in the late 90's, it seems that they did not made their way in the information retrieval community. However, they have shown good performances in the case of information retrieval with mobile devices where the network capabilities are unreliable in nature [16]. There are good reasons for using mobile agents in the distributed CBIR context, such as the reduction of the network load (the processing code of the agent being very small in comparison to the feature vector indexes) and the massive parallelization of the computation. Some of these advantages are described in [17].

The resulting scenario of CBIR using mobile agents is to launch several mobile agents with a copy of the query. They crawl the network in search of image collections. When an agent gets to a site, a dialog with a local agent is established. The local agent indexes the images and performs the processing [18].

D. Ant-like agents

In addition to the computational interest of mobile agents, we suggest that they can also participate to the learning process. Software agents following ant-like behaviors have been used, for instance, in several domains including network routing, traveling salesman problem or quadratic assignment problem [19] following the model of the ethologist J. L. Deneubourg [20]. In the case of distributed retrieval, ant-agents crawl the network to find the relevant documents. They move from one peer to another and mark the visited hosts (by changing a numerical value locally stored on these hosts, called *marker*). These markers can be viewed as a collective memory of paths leading to the relevant sites. This behavior-based mapping of the network is well adapted to inconsistent networks such as peer-to-peer networks, since the marked paths evolve with the global trend of the agent movements [21], [22]. In our distributed CBIR context, we have to do several travels between the user's computer and the information sources. Ant-algorithm seems to be a good solution for learning the relevant paths through the dynamics of active learning.

III. Architecture of our system

Fig. 1 describes our system. The user starts his query by giving an example or a set of examples to an interface (1). We build a similarity function based on these examples (2). Mobile agents are then launched with a copy of this similarity function (3). Every host of the network contains an agent platform in order to be able to receive and execute incoming mobile agents. These agents travel through the network according to movement rules described in the next section (4). These rules allow our system to learn the paths leading to relevant images.

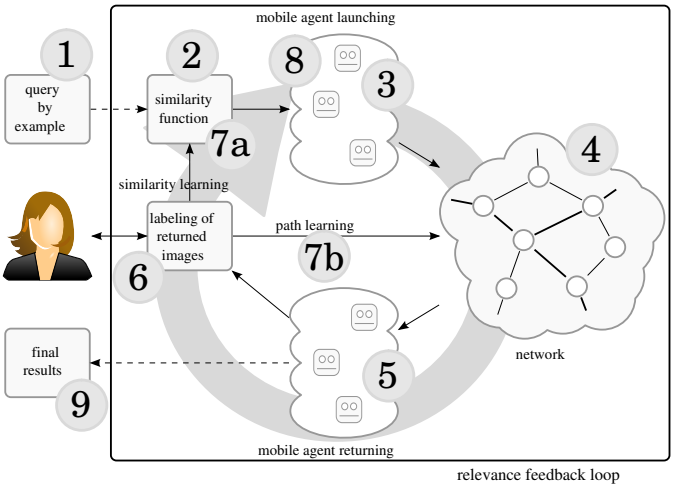


Fig. 1. Functional description of our system showing the user in interaction with the relevance feedback loop (launching of agents, retrieval, display and labeling).

On each platform, an agent indexing the local images is run. The visual feature vectors used consist of color and texture distributions. The colors are obtained from the quantization into 32 bins of the *Lab* color space, while the textures are obtained from the quantization into 32 bins of the output of 12 Gabor filters. The quantization is performed on a set of common images. Both color and texture distributions are normalized over all images regarding each bin. The resulting

image descriptions are vectors concatenating the color and texture distributions, mapping the images to a feature space of dimension 64. The incoming mobile agent sends the similarity function to the index agent which returns the most informative images regarding active learning.

In our context, the pool of unlabeled documents is split into several sets, stored on the different hosts of the network. How can we find the examples that will enhance the most the similarity function in this context? We divide the selection into two stages: first, we design a strategy to select relevant collections and then we choose the local examples to be labeled. The collection selection stage involves an ant-like algorithm ruling the agent movements in order to retrieve images from well-chosen collections. Agents increase and decrease the level of markers of each host they visit: hosts containing a lot of relevant images will have a high level of marker, whereas hosts with no relevant images will have a low level of markers (as described in section IV). The local selection stage is done by an active learning strategy adapted to the mobile agents context and based on uncertainty sampling, as described in section V. A scenario of this distributed active learning process is shown on Fig. 2.

As soon as they receive the answer of the index agent, the mobile agents return to the user's computer (5) and the results are displayed on the interface (6). The user can label these results (1: *relevant*, -1: *irrelevant*), and the similarity function is updated consequently (7a) as well as the good paths of the network (7b). As the similarity function we use is based on SVM analysis [23], the update only consists in adding the results and their labels to the training set and to train a new SVM. Mobile agents are then relaunched with the improved similarity function. The interactive loop consists in several launching of mobile agents and labeling of the results (8). At the end of the interaction, mobile agents are launched for a very last time in order to retrieve the best results from each host (9). The number of retrieved images is proportional to the level of the markers leading to this host. In combination with the ant-algorithm, this assures that most of the best retrieved images are provided by relevant hosts.

IV. Ant-like algorithm

Agents move following an ant-inspired algorithm as described in [24], [25]. We improved the classical reinforcement rules by adding a semantic-oriented reinforcement taking into account the interaction with the user.

A. Algorithm description

Let c be the host currently executing the agent and S the set of possible destinations from c. Each host i of S contains a marker ph_i which is used by the agent to determine the next move. Just like ants with pheromones, the higher the level of marker ph_i the higher the probability P_i to move to the host i:

$$P_i = \frac{ph_i}{\sum_{k \in S} ph_k} \tag{1}$$

While homing with retrieved images, agents increase the level of the marker on the visited hosts, given the following rule:

$$\frac{\Delta ph_i}{\Delta t} = +\beta \cdot a(t) \tag{2}$$

with $a(t)$ (*agent reinforcement*) being 1 when the agent has found a collection and 0 else, and t increasing if and only if the marker of the concerned host is changed. Therefore, t is a time only related to the concerned host. The system reinforces paths leading to informative hosts, since this increase of markers will also increase the probability of these hosts to be visited later.

While searching for images, agents decrease the level of the marker of the destination, with the following rule:

$$\frac{\Delta ph_i}{\Delta t} = -\alpha \cdot ph_i(t) \tag{3}$$

This allows to forget the non informative paths. Compared to real ants, this rule models the evaporation of pheromones.

B. Our implementation

As a marker is only updated when an agent moves towards the host holding it, an increase of time t represents a travel of an agent along a specific path. Thus, the time t is local to a host of the network, depending on how many agents are moving through it. If no agent moves along a path, the local time is frozen and markers do not evolve anymore. On the contrary, if a lot of agents move along another path, the related time flies very fast and markers are updated very quickly. One of the main consequence is that there is no correlation between the dynamics of the times of two hosts that are not bound to each other. The system does not forget a path just because no agent travels through it. One main advantage of a local time is that the system does not need to synchronize all the hosts of the network with an universal clock. Each host has its internal clock based on the frequency of agent visits.

Each time the user labels an image, the levels of markers on the path taken to retrieve this image are also increased as follows:

$$\frac{\Delta ph_i}{\Delta t} = +\gamma \cdot u(t) \tag{4}$$

Where $u(t)$ is the reinforcement signal given by the user (*user reinforcement*): 1 if the label is positive, 0 otherwise. Thanks to this rule, the paths leading to hosts containing images the user is looking for are reinforced. The global equation may be written as follows:

$$\frac{\Delta ph_i}{\Delta t} = -\alpha \cdot ph_i(t) + \beta \cdot a(t) + \gamma \cdot u(t) \tag{5}$$

When the variations of ph_i tend to zero, the estimation $\hat{ph}_i$ of the marker's level is:

$$\hat{ph}_i = \frac{\beta \cdot \hat{a} + \gamma \cdot \hat{u}}{\alpha} \tag{6}$$

With $\hat{a}$ and $\hat{u}$ being the estimation of a and u respectively. In that case, the level of markers tends to zero if the host does not lead to a collection, and is not null on the contrary.

The highest level is obtained for a host with high values of $\hat{u}$ (which implies $\hat{a} = 1$), that is on a path leading to a collection containing a lot of relevant images.

C. Re-use of marker - long term learning

We have also implemented a very simple way to keep in memory the previously learned collection selection. In this case, we do not reset the markers at the beginning of each new search session. The markers evolve from a query to another leading to a reinforcement of the collections which gave the greatest number of positive images over all the sessions. If a collection contains too few positive images and is exhausted before the end of the current session, its markers will be very small for the next search. Thus, only collections with large sets of positive images will be reinforced with this strategy. However, these small veins of positives images are not lost: they may be rediscovered by an agent at any time, due to the stochastic behavior of our system.

V. Distributed active learning

A. Local images selection

Each time an agent gets to a site containing a collection, it has to choose some examples to add to the training set. Given a set of images $\{\mathbf{x}_k\}$ and their corresponding labels $\{\mathbf{y}_k\}$, let us denote a training function τ giving the label $\mathbf{y}$ to the image $\mathbf{x}$. The aim of active learning is to choose the unlabeled image $\mathbf{x}$ that will enhance the most the relevance function when added to the previous set $\{\mathbf{x}_k\}$. The label $\tau(\mathbf{x})$ is correspondingly added to the previous labeling set $\{\mathbf{y}_k\}$. We use uncertainty-based sampling, which is the most used active strategy in image retrieval [26]. This strategy aims at selecting an unlabeled image that the training function τ is most uncertain about. The first solution is to compute a probabilistic output for each image, and select the unlabeled images with the probabilities of being relevant closest to 0.5 [27]. Similar strategies have been also proposed with SVM classifier [28], where the similarity function is the distance to the hyperplane, and the most uncertain documents have an output close to 0.

As many agents reach the same host with the same relevance function, the active strategy should not answer in a deterministic way, otherwise all these mobile agents will get the same answers, and thus act as one single agent. We recently proposed an active learning scheme [29] to boost the retrieval in one single collection. We propose here to extend our active learning framework in order to handle datamining over a network. We rank images given their distance to the boundary. We divide the selection of I images into I selections of a single image. Each of these selections are done over a set of images with ranking between 1 and n using an uniform distribution. The selected image is then removed from the set. To fix the width of the set to n for the i-est selection, we add the image ranked $n + i - 1$ to the selection pool. Let us denote $p = \frac{1}{n}$ the probability of an image $\mathbf{x}$ in the set at round i of being selected, and $q = 1 - p$. $P_i(\mathbf{x})$ the probability for the image $\mathbf{x}$ with rank $r_{\mathbf{x}}$ to be exactly selected at round i follows a specific geometric distribution depending on the round where it happened:

$$P_i(\mathbf{x}) = \begin{cases} p\prod_{j=1}^{i-1} q & , \quad 1 \leq r_{\mathbf{x}} \leq n \\ p\prod_{j=1}^{i-r_{\mathbf{x}}+n-1} q & , \quad n+1 < r_{\mathbf{x}} < n+i \\ 0 & , \quad n+i \leq r_{\mathbf{x}} \end{cases} \tag{7}$$

$$P_i(\mathbf{x}) = \begin{cases} pq^{(i-1)} & , \quad 1 \leq r_{\mathbf{x}} \leq n \\ pq^{(i-r_{\mathbf{x}}+n-1)} & , \quad n+1 < r_{\mathbf{x}} < n+i \\ 0 & , \quad n+i \leq r_{\mathbf{x}} \end{cases} \tag{8}$$

The probability of an image being selected after I rounds is the sum of the probabilities of being selected at each round from the round where it has entered the set:

$$P(\mathbf{x}) = \begin{cases} \sum_{i=1}^{I} pq^{(i-1)} & , \quad 1 \leq r_{\mathbf{x}} \leq n \\ \sum_{i=r_{\mathbf{x}}-n+1}^{I} pq^{(i-r_{\mathbf{x}}+n-1)} & , \quad n < r_{\mathbf{x}} < n+I \\ 0 & , \quad n+I \leq r_{\mathbf{x}} \end{cases} \tag{9}$$

Which can be rewritten using the geometric series of parameter q as :

$$P(\mathbf{x}) = \begin{cases} 1 - q^I & , 1 \leq r_{\mathbf{x}} \leq n \\ 1 - q^{(I-r_{\mathbf{x}}+n)} & , n < r_{\mathbf{x}} < n+I \\ 0 & , n+I \leq r_{\mathbf{x}} \end{cases} \tag{10}$$

The probability P follows an uniform distribution for images with ranking between 1 and n. For images with ranking between n and $n+I$, it decreases exponentially with parameter $q = 1 - \frac{1}{n}$. It is null otherwise. This selection strategy combines the uncertainty based sampling with a diversity introduced by the distribution P. It is also very fast to compute (one ranking of all images plus I random sampling following a uniform distribution) using the following algorithm:

1) Rank all images given their distance to the hyperplane
2) Add the n first to an empty pool
3) for i from 1 to I do
   Randomly select an image from the pool using uniform distribution
   Remove it from the pool
   Add the image $n + i$ to the pool
4) done

B. Collection selection

In our context, the relevant category is very little in front of the available data. Thus, a relevant image might often be considered as more informative than an irrelevant one. We consider a good collection selection strategy the one that selects the collections containing mainly relevant images. Our collections selection strategy is performed by the ant algorithm. At the very beginning, all collections have an equal chance of being visited by an agent (all markers are set to 1).

After a few rounds, thanks to the user's reinforcement $u(t)$ (Cf. Section IV), the highest probability of selecting images will be obtained for the collection that returned the largest set of positively labeled images.

In case of a collection with only few positive images, this strategy will at first reinforce the selection on this collection. But as soon as the vein is exhausted, no more positive labels will reinforce the marker, and the probability will decrease quickly. This shows how the strategy adapts through the dynamics of active learning.

VI. Experiments.

A. Experiment setup

While we have only made simulations of the network on the COREL dataset in [30], we did a real implementation of our system for the experiments presented hereby. To test our system in quasi "realistic" conditions, we performed the experiments on the intranet of our lab. In a previous work it has been tested that, given the path reinforcement strategy, no matter the complexity of the network is, it is reduced, after learning, to a simple connexion graph with only informative paths remaining (in accordance with [24]). Indeed, as our algorithm reinforces the shortest path to the information sites, it can then be considered only a small number of directions (some of which leading to relevant images), no matter how many computers are in these directions. We made our experiments with four directions, each one consisting of a single computer. For the data, we used the set of keyframes of the $trecvid'05$ competition for a total of about 75000 images. We used as ground truth the high level features given by CMU. We divided these images into four collections, each hosted by a bi-opteron 275 with 8 GB of RAM. These hosts were the possible destinations for our mobile agents. The links between the hosts of the network were 100Mbps, and therefore the time taken by the agents to migrate was negligible.

For the agent framework, the *Jade*[2] platform was used. As *Jade* had little support for agent mobility between several platforms, we implemented our own mobility service using Java serialization and ACL messaging. As the agent code was common to all platforms, the serialization of an agent resulted in a string containing only its internal fields. The size of the messages containing a serialized agent was about *10kB*. The size of the similarity function depended on the number of examples in the training set, and varied from less than *1kB* to *30kB*. This made the serialized agents as light as a typical web page with few images, such as the page containing the results of a google search.

B. Network learning

The main assumption for learning paths to relevant images together with the relevance function is that the images belonging to the concept are well localized on the network. In other words, there are hosts that contain a lot of relevant images and hosts that do not contain any or few relevant images. To simulate this, we hosted in a first case all the relevant images on the first destination, while the other computers did not contain any relevant image. We called this setup *strong localization*. In a second case, we put only 80% of the relevant images on the first host, the remaining being equally distributed on the other hosts, in order to simulate what we called a *weak localized* setup. In order to deal with a realistic retrieval where the relevant images are hidden in a mass of irrelevant images, we added to each host about 15000 irrelevant images randomly selected. 100 search sessions were made for each category, and for each new query, the markers were reset. Due to the ant-algorithm, we expected at the end of the retrieval session the levels of marker being high on the first host compared to the other ones. Fig. 3 shows the level of markers for each host for the case where relevant image are localized at 100% on the first destination. As we can see, we were able to learn the good path, with relative success from a category to another. Actually, as the learning process depends on the user's reinforcement $\hat{u}$, it is directly linked with the number of relevant images found during the interaction, which is very dependent on the category.

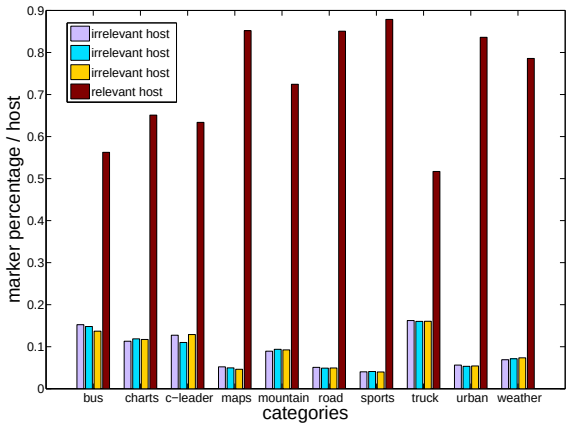


Fig. 3. Relative proportions of markers on the four hosts per category. A bar at 70% means that an agent has a probability of 70% to go to this host. The destination containing the relevant images (name *relevant host*) has always the highest probability of being visited.

For the weak localization, Fig. 4 shows that learning the relevant path is also a success even if there are some non-zero value of $\hat{u}$ for the other hosts. Nevertheless, this is less pronounced than for the strong localization setup.

C. Similarity learning

In order to evaluate the influence of path learning on category learning, we also ran experiments of our active learning strategy as described in section V-A on a single bi-opteron 275 containing the whole image collection. Due to the lack of parallelization, this setup took a much longer time than our mobile-agents system. This setup will be referred as *centralized setup* in the following. Fig. 5 shows the *recall@500* (number of relevant images retrieved for 500 total images retrieved) per setup for each category. In order to compute the recall, the number of the best images retrieved from each host were proportional to the level of marker.

[2]http://jade.tilab.com

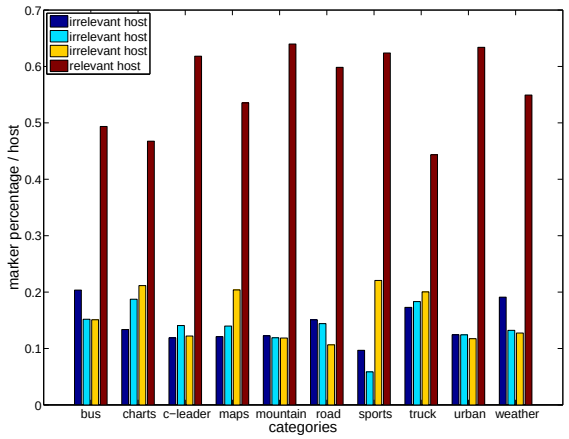


Fig. 4. Relative proportions of markers on the four hosts per category for the *weak localization* setup. The host containing most of the relevant images has always the highest probability of being visited, despite it is less pronounced than for the *strong localization* setup.

For instance, if the levels were $\{0.80, 0.10, 0.05, 0.05\}$, we retrieved $\{400, 50, 25, 25\}$ images from each host respectively. In the case of the *centralized setup*, the recall was made among the 500 best images of the whole collection.

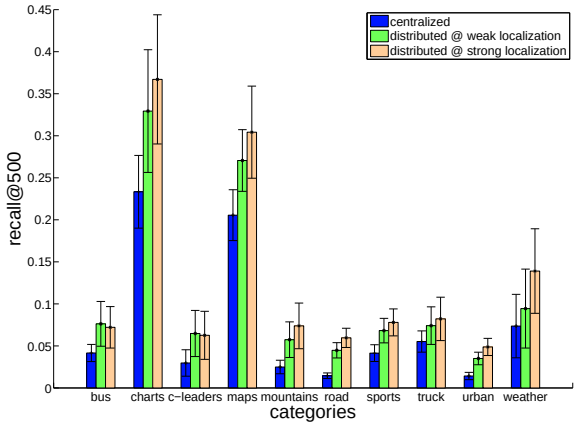


Fig. 5. $Recall@500$ for the centralized setup and the distributed setups with both strong and weak localization. Our distributed implementation outperforms the centralized setup, even in case of weak localization.

We can see that the distributed system outperforms the centralized one in all cases, but especially for difficult categories (low recall values, for instance *'road'* or *'urban'*). The standard deviations for these varied from 1% for difficult categories (less than 10% in recall) to about 5% for easy categories (more than 20%).

D. Re-use of markers

As explained in Section IV-C, We ran the same experiments without resetting the markers level at the beginning of a search. For each category, the initialization of the markers was made only once for the first query. Fig. 6 shows the levels (mean over the sessions) of markers for each category. The levels of paths leading to the irrelevant hosts are negligible, which means that almost all agents move towards the first host (but not all due to the algorithm non-determinism).

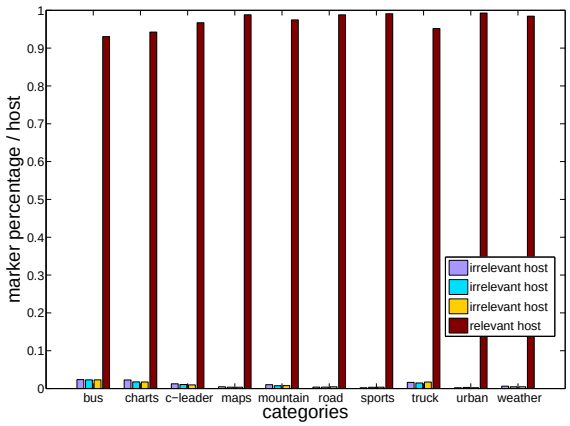


Fig. 6. Relative proportions of markers on the four hosts per category for the *re-use of markers* setup. The levels of irrelevant hosts are negligible.

The *recall@500* of the Fig. 7 shows that this setup is a further improvement. For easy categories (for instance *'charts'* or *'maps'*), the gain is up to about 100%. For difficult categories (*'road'*), the gain can be as high as about 1000%. As this strategy focuses even more on adding relevant images

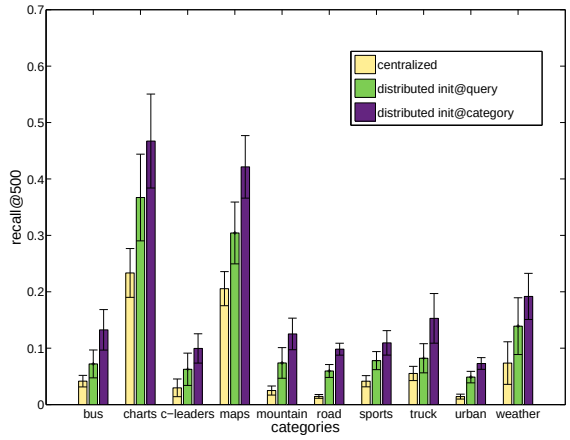


Fig. 7. $Recall@500$ per category comparing the $centralized$ setup, the distributed setup and the *re-use of markers* setup. The recall is at least 50% better than the distributed setup, and at least 100% better than the centralized setup.

to the training set (by keeping in memory their localization over many sessions), this can explain the boost obtained with difficult queries. This is shown on Fig. 8, where the number of positive labels per session per category is clearly higher than for the other setups, even if the standard deviation is high (about 10 labels).

These experiments validated our path learning algorithm by showing that the path leading to the relevant host has the highest level of marker. We then presented results showing that taking in account the localization of the relevant images in

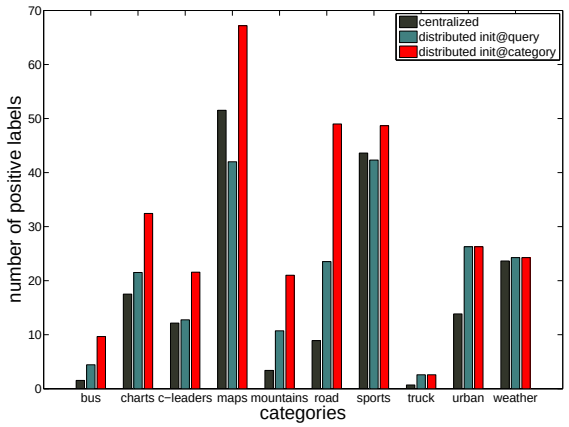


Fig. 8. Mean number of positives labels obtained in a session. The *re-use of markers* setup obtained more positives labels during the interaction than the centralized and the distributed setup.

the learning process leads to an improvement of 75% in mean. Moreover, in a setup where the markers were not initialized at the beginning of each new search but kept for the next session, our system had an improvement of about 155% in mean.

VII. Conclusion.

In this article, we presented a new active learning strategy for searching images over networks. We introduced a new reinforcement based learning scheme for learning the localization of relevant images. We carried out a smart cooperation between these two strategies in a global architecture based on mobile agents with an ant-like behavior. We made a working implementation of our system and tested it on a real network using the *trecvid* keyframe dataset. It shows that our system is definitively an improvement to distributed CBIR.

We are now working on an extension of our system for managing different markers, each one related to a specific concept. Concurrent queries of different concepts will involve different markers. With an efficient management of many concept-dependant markers, we believe the improvement shown in the re-use of markers setup can be widely used.

References

[1] J. Cho and S. Roy, "Impact of search engines on page popularity," in *WWW '04: Proceedings of the 13th international conference on World Wide Web*. New York, NY, USA: ACM, 2004, pp. 20–29.

[2] R. Veltkamp, "Content-based image retrieval system: A survey," University of Utrecht, Tech. Rep., 2002.

[3] M. Wood, N. Campbell, and B. Thomas, "Iterative refinement by relevance feedback in content-based digital image retrieval," in *ACM Multimedia 98*, Bristol, UK, September 1998, pp. 13–20.

[4] T. Huang and X. Zhou, "Image retrieval with relevance feedback: From heuristic weight adjustment to optimal learning methods," in *International Conference in Image Processing (ICIP'01)*, vol. 3, Thessaloniki, Greece, October 2001, pp. 2–5.

[5] C. Carson, M. Thomas, S. Belongie, J. Hellerstein, and J. Malik, "Blobworld: A system for region-based image indexing and retrieval," in *Third Int. Conf. on Visual Information Systems*, June 1999.

[6] D. Lowe, "Distinctive image features from scale-invariant keypoints," in *International Journal of Computer Vision*, vol. 20, 2003, pp. 91–110.

[7] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin, "The QBIC project: Querying images by content, using color, texture, and shape," in *Storage and Retrieval for Image and Video Databases (SPIE)*, February 1993, pp. 173–187.

[8] S. Santini, A. Gupta, and R. Jain, "Emergent semantics through interaction in image databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 3, pp. 337–351, 2001.

[9] N. Vasconcelos, "Bayesian models for visual information retrieval," Ph.D. dissertation, Massachusetts Institute of Technology, 2000.

[10] O. Chapelle, P. Haffner, and V. Vapnik, "Svms for histogram based image classification," *IEEE Transactions on Neural Networks*, vol. 9, 1999.

[11] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification." *Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2001.

[12] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, 2002.

[13] A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349–1380, December 2000.

[14] S. Berretti, A. D. Bimbo, and P. Pala, "Merging results for distributed content based image retrieval." *Multimedia Tools Appl.*, vol. 24, no. 3, pp. 215–232, 2004.

[15] I. King, C. H. Ng, and K. C. Sia, "Distributed content-based visual information retrieval system on peer-to-peer networks," *ACM Trans. Inf. Syst.*, vol. 22, no. 3, pp. 477–501, 2004.

[16] Y. Jiao and A. R. Hurson, "Performance analysis of mobile agents in mobile distributed information retrieval system - a quantitative case study." *Journal of Interconnection Networks*, vol. 5, no. 3, pp. 351–372, 2004.

[17] D. B. Lange and M. Oshima, "Seven good reasons for mobile agents," *Commun. ACM*, vol. 42, no. 3, pp. 88–89, 1999.

[18] V. Roth, U. Pinsdorf, and J. Peters, "A distributed content-based search engine based on mobile code," in *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*. New York, NY, USA: ACM Press, 2005, pp. 66–73.

[19] E. Bonabeau, M. Dorigo, and G. Theraulaz, "The social insect paradigm for optimization and control," *Nature*, vol. 406, pp. 39–42, 2000.

[20] J. Deneubourg, S. Goss, N.Franks, A. Sendova-Franks, C. Detrain, and L. Chrétien, "The dynamics of collective sorting: Robot-like ants and ant-like robots," in *From Animals to Animats: Proc. First Int. Conference on Simulation of Adaptive Behavior*, J.-A. Meyer and S. Wilson, Eds., Paris, France, 1990, pp. 356–363.

[21] A. Revel, "Web-agents inspired by ethology: a population of "ant"-like agents to help finding user-oriented information." in *IEEE WIC'2003 : International Conference on Web Intelligence.*, IEEE. Halifax, Canada: IEEE Computer Society, October 2003, pp. 482–485.

[22] ——, "From robots to web-agents: Building cognitive software agents for web-information retrieval by taking inspiration from experience in robotics," in *ACM International conference on Web Intelligence*, Université Technologique de Compiègne, Compiègne, France, Septembre 2005.

[23] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[24] M.Dorigo, V. Maniezzo, and A. Colorni, "The ant system: Optimization by a colony of cooperating agents." *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, vol. 1, no. 26, pp. 29–41, 1996.

[25] E. Bonabeau, M. Dorigo, and G. Theraulaz, "Inspiration for optimization from social insect behaviour," *Nature*, vol. 406, pp. 39–42, 6 July 2000.

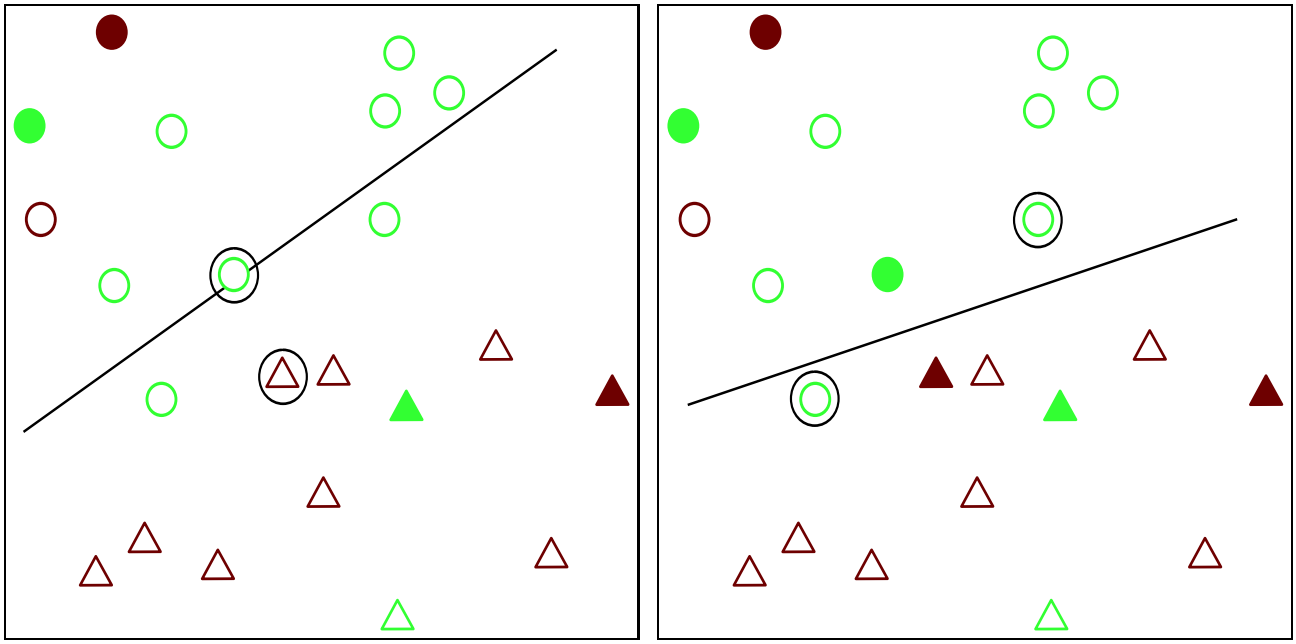
[26] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *ACM Multimedia*, 2001.

[27] D. Lewis and J. Catlett, "Heterogenous uncertainty sampling for supervised learning," in *International Confenrence on Machine Learning*, 1994.

[28] J. Park, "On-line learning by active sampling using orthogonal decision support vectors," in *IEEE Neural Networks for Signal Processing*, 2000.

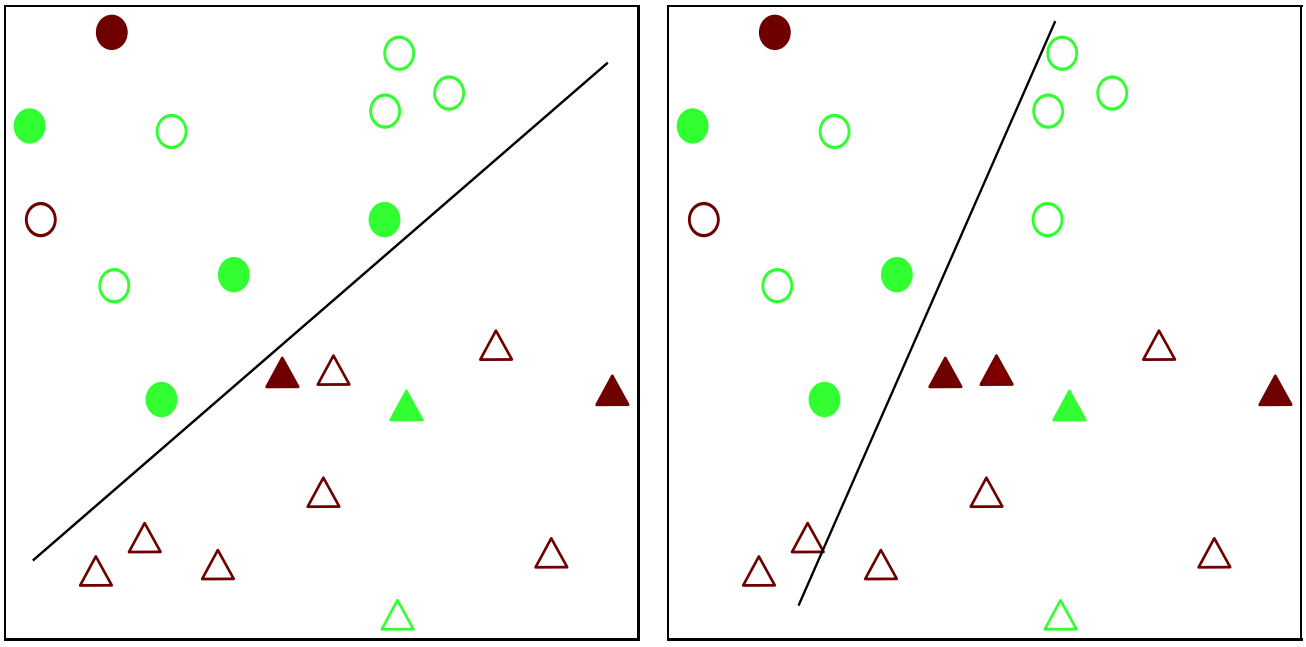
[29] M. Cord, P.-H. Gosselin, and S. Philipp-Foliguet, "Stochastic exploration and active learning for image retrieval," *Image and Vision Computing*, vol. 25, pp. 14–23, 2007.

[30] D. Picard, M. Cord, and A. Revel, "Cbir in distributed databases using a multi-agent system," in *IEEE International Conference on Image Processing (ICIP'06)*, Atlanta, GA, USA, October 2006.



(a) The first round of active learning will select two images to be labeled. The most uncertain (closest to the boundary) from each collection is chosen since we don't have any information about the images each collection contains.

(b) As there are more positive labels on A than on B, the next active learning round will select the two most uncertain images only on A.



(c) The online collection selection leads quickly to an efficient classification by improving (uncertainty based) the classification only where there are relevant images.

(d) If there is no DB selection algorithm, the two selected examples (one from each DB) would lead to a less efficient classification.

Fig. 2. Example of distributed active learning with two collections A (bright green color) and B (dark red color). A contains many relevant objects, whereas B has very few. Plain symbols denote labeled images, whereas hollow symbols denote unlabeled images. The initial query contains two relevant images (circles) and two irrelevant images (triangles) equally chosen from A and B. The collection selection algorithm will choose A since the active learning returns more positive labels on A.