Collaborative Classifier Agents: Studying the Impact of Learning in Distributed Document Classification

Conference Paper · June 2007

DOI: 10.1145/1255175.1255263 · Source: DBLP

CITATIONS
9

READS
286

3 authors, including:

Weimao Ke
Drexel University
69 PUBLICATIONS **1,357** CITATIONS

SEE PROFILE

Collaborative Classifier Agents: Studying the Impact of Learning in Distributed Document Classification

Weimao Ke
Laboratory of Applied Informatics Research
Indiana University, Bloomington
1320 E. 10th Street, LI 011
Bloomington, Indiana
47405-3907
wke@indiana.edu

Javed Mostafa
Laboratory of Applied Informatics Research
Indiana University, Bloomington
1320 E. 10th Street, LI 011
Bloomington, Indiana
47405-3907
jm@indiana.edu

Yueyu Fu
Laboratory of Applied Informatics Research
Indiana University, Bloomington
1320 E. 10th Street, LI 011
Bloomington, Indiana
47405-3907
yufu@indiana.edu

ABSTRACT

We developed a multi-agent framework where agents had limited/distributed knowledge for document classification and collaborated with each other to overcome the knowledge distribution. Each agent was equipped with a certain learning algorithm for predicting potential collaborators, or helping agents. We conducted experimental research on a standard news corpus to examine the impact of two learning algorithms: Pursuit Learning and Nearest Centroid Learning. For a fundamental retrieval operation, namely classification, both algorithms achieved competitive classification effectiveness and efficiency. Subsequently, the impact of the learning exploration rate and the maximum collaboration range on classification effectiveness and efficiency were examined. Close investigation of agent learning dynamics revealed increasing and stabilizing patterns that were enhanced by the learning algorithms.

Categories and Subject Descriptors

H.3.3 [**Information storage and retrieval**]: Information Search and Retrieval—*Information filtering*; H.3.4 [**Information storage and retrieval**]: Systems and Software—*Distributed systems*; H.3.4 [**Information storage and retrieval**]: Systems and Software—*Information networks*; H.3.7 [**Information storage and retrieval**]: Digital Libraries—*Systems issues*

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

Information retrieval, distributed classification, multi-agent system, learning, collaboration, effectiveness, efficiency

JCDL'07, June 17–22, 2007, Vancouver, British Columbia, Canada.

1. INTRODUCTION

Classification, a knowledge organization mechanism, is a way we humans understand the world by aggregating like-entities [10, 28]. "The ability to classify is an essential part of life [24, page 119]" Document Classification, or Categorization, has been a research area in Machine Learning (ML) and Information Retrieval (IR) and is relevant to information extraction and knowledge discovery [8, 23]. It is a fundamental function of IR and can be applied to various digital library processes such as indexing and filtering [24, 15, 7]. Traditional classification approaches assume that global knowledge is available at a centralized place. However, this assumption is rarely true in the real world. Evidence is emerging that, given the distributed nature of knowledge, collaboration is the main force driving knowledge management/discovery and making possible the emergence of a global brain [26, 21, 2]. The World Wide Web is a good example of knowledge distribution, where Web sites serve narrow information topics and tend to form communities through hyperlinks [4].

Distributed IR has become a fast-growing research topic in recent years. Recent distributed IR research has been focused on intra-system retrieval fusion, cross-system communication, decentralized P2P network, and distributed information storage and retrieval algorithms [3]. Research also concentrated on genetic algorithms for feature selection, intelligent crawling, information routing, etc. Modeling agent collaboration for document classification, however, has drawn little attention from distributed IR researchers. Related work, although limited, indicated potential for success in this area. In a comparison between single-agent and multi-agent classification, [19] discussed several advantages such as fault tolerance, adaptability, flexibility, resource sharing, privacy, and economics.

One previous work used adaptive distributed agents for crawling distributed information [13]. Following the same direction, another research study employed distributed agents to traverse links in an encyclopedia and answer user questions; the investigators studied the influence of learning in the adaptive retrieval process [20]. The other studies used collaborative agents to achieve better filtering and classification performance collectively while presenting key issues in designing a multi-agent framework for data mining and classification [30, 25].

In recent years, Mukhopadhyay et al. examined application of distributed agents to information filtering and classification tasks [16, 17]. Their research using an Acquaintance List learning algorithm and other collaboration strategies showed that learning helped distributed agents do classification better without consuming too much communication resources. However, their experimental results in terms of classification effectiveness were not sufficiently competitive. Another research study led to different collaborative learning algorithms and produced high classification effectiveness but failed to measure efficiency [14, 5].

1.1 Research Problem and Questions

Document classification is a fundamental problem in IR and, hence, by extension, in digital libraries. We found automated techniques for classification, particularly in the distributed Web setting, is an understudied area and there are several key challenges that require closer scrutiny and study.

In this paper, we will describe a study examining the impact of learning in a distributed environment where agents learn to collaborate with each other for document classification. We are particularly interested to know: to what extent learning can improve classification effectiveness without sacrificing too much efficiency and what the differences are among non-learning, random learning, and adaptive learning algorithms. We also examine the characteristics of learning dynamics, e.g., classification effectiveness over time.

2. DISTRIBUTED DOCUMENT CLASSIFICATION

In our research, the knowledge for classification is represented by a set of terms we refer to as class-term vectors (we elaborate on the vector structure below). A centralized classifier has global knowledge as all the class-term vectors are maintained in one place. By dividing the class-term vectors into small subsets and distributing them among multiple agents, we are able to create a decentralized environment where each agent has partial knowledge.

To study document classification, we first consider traditional classification methods, which are still applicable to a distributed environment. These methods, described below, include document representation and classification algorithms. Then, we introduce information distribution and discuss its influence on document classification.

2.1 Representation of Documents

For document representation, we use the widely used Vector-Space Model [22] to construct document-term vectors. Feature selection produces a thesaurus for a document collection. This thesaurus is then used to represent each document using the TF*IDF (term frequency * inverse document frequency) weighting scheme. A document is then converted to a numerical vector where the i^{th} item is computed by:

$$W_i = T_i * log(N/n_i) \quad (1)$$

where T_i is the frequency of the i^{th} term of the thesaurus in the new document, N is the total number of documents in the representative document set, and n_i is the number of documents in the representative document set containing the i_{th} term of the thesaurus. TF*IDF is a well-known and effective technique for term weighting [1].

2.2 Classification of Documents

To classify documents, we use a similarity measure based on term vectors. Document-term vectors are produced by the TF*IDF representation scheme described above while each class is represented using the centroid of training documents prelabeled to the class. Then a similarity measure, called Cosine Similarity Coefficient [9, 1], is used to compute the cosine of an angle between two vectors. Given two non-null vectors $X = [x_1, .., x_t]^T$ and $Y = [y_1, .., y_t]^T$, their cosine similarity is computed by:

$$\sum_{i=1}^{t} x_i y_i / \sqrt{(\sum_{i=1}^{t} x_i^2)(\sum_{i=1}^{t} y_i^2)} \quad (2)$$

During classification, a classifier compares each document with the available classes by computing their similarity values. Then the classifier chooses the class with the maximum value and applies a threshold to determine whether the value is high enough for the document to be classified to that class.

2.3 Distributed Knowledge and Collaborative Classification

In a distributed environment, where only a subset of the classes is available, a classifier (i.e. an agent) chooses the top class in the subset and does not have global knowledge to determine if the class is the most relevant or not. As knowledge of each agent becomes limited in such a distributed environment, the agents will not achieve high classification quality if they work alone. They have to collaborate with each other–to seek help from other agents when one fails. In this study, the proposed agent collaboration model is illustrated in Figure 1.

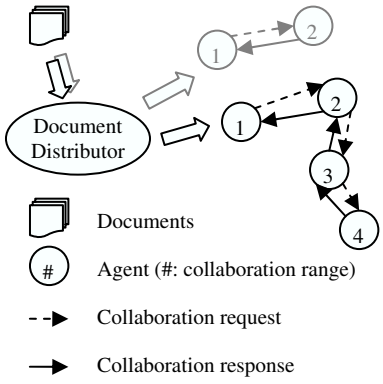

Figure 1: Agent collaboration network

As shown in Figure 1, the document distributor loads the document collection and releases one document to a randomly selected classifier agent each time. The chosen agent takes the incoming document and tries to classify it. If it fails to classify the document locally (i.e. the maximum similarity score it has achieved is lower than a predefined threshold), it asks another agent for help by sending the document to the remote agent. If the remote agent succeeds in classification, it sends the result back. Otherwise, it asks

another agent for help if only the *range* of collaboration is smaller than a pre-defined maximum number (i.e. Maximum Collaboration Range g, which we will discuss in detail), and so forth. Only after the classification result is back will the document distributor release another document.

The agents know nothing about their environment initially and learn to collaborate gradually. In this way, we are able to observe the adaptability of the learning algorithms. To avoid overdependence of agents on some neighbors, we assign a random factor called Exploration Rate r to the learning processes. In terms of the rate, an agent tries random neighbors occasionally without relying on its knowledge of neighbors who assisted in the past. We discuss this factor in detail later.

We use the streaming model described above instead of letting all classifier agents do classification tasks simultaneously. We simply discuss the main reasons here. Firstly, as the document distributor distributes each document to a randomly chosen agent, all agents have equal opportunities to do the tasks. Secondly, in this way, we are able to track agent collaborations sequentially. This is especially necessary for learning convergence analysis. Thirdly, each document that has gone through the agent collaboration network will train the agents so that they might do better with documents coming afterwards. This increases the probability for a learning algorithm to converge. Depending on the learning algorithm, an agent may get to know the agent society after several iterations of collaborations and learn to collaborate in a more effective and/or efficient way. Below we present a discussion on learning algorithms we developed for collaborative classification.

3. LEARNING IN THE DISTRIBUTED ENVIRONMENT

3.1 The Necessity of Learning

The distribution of knowledge requires agents to collaborate with each other and make possible the emergence of global knowledge. For an agent to collaborate "intelligently", learning from its collaboration memory/history and understanding the environment is necessary. Moreover, the diversity of the documents brings about dynamic changes to the classification tasks. Therefore, adaptive learning will be helpful for the agents to keep up with such changes and deliver consistent service. Note that the "learning" here refers to learning about peer agents, i.e. learning to collaborate, instead of learning about document classes and learning to classify into those classes.

In this study, we propose two learning algorithms. One algorithm is based on a stationary view of the environment, i.e. an agent that has helped a lot will continue to help regardless of changes in document content. The other algorithm is content-sensitive and builds correlations between document contents and the helping agents through collaboration and learning. We will compare the two algorithms with non-learning and random learning, and study the impact of the Exploration Rate r and the Maximum Collaboration Range g.

3.2 Learning Algorithm: Pursuit Learning

There have been a variety of reinforcement learning algorithms proposed for different environments and learning strategies. The reinforcement learning algorithm used in this paper is called Pursuit Learning, which is a model-based algorithm and assumes a stationary environment [18]. The learning is not content-sensitive as it disregards the diversity of document contents streamed at each agent. A description of this algorithm is given below.

Suppose an agent has a finite number of actions, i.e. the number of its neighboring agents. It performs one of the actions and receives a certain reward because of the action. Given an action probability vector $P = [p_1, ..., p_n]^N$, where each value p_i denotes the probability with which the agent chooses action α_i and is set to 1/n initially. Let $d = [d_i]$ denote the underlying reward probabilities vector while $\hat{d}$ is the estimate of the d vector. $\hat{d_i}$ is the estimated probability that action α_i results in a reward, which is the cumulative average of the reward values for each action. The learning agent maintains and updates the p and $\hat{d}$ vectors of dimensions equal to the number of actions. α_m is the unknown optimal action (i.e., $d_m = max_{i=1}^{n} d_i$). S_i is the number of times action α_i has been tried so far. The desired behavior is that p_m should be approaching 1, all other p_i's should approach zero. At instance k, choose an action in terms of vector P and the exploration rate r:

91: $rnd \leftarrow Randomize(1.0)$
92: **if** $rnd < r$ **then** {The agent "explores"}
93: Select an action randomly regardless of vector P
94: **else** {The agent predicts based on learned knowledge}
95: Select an action that has the maximum value in vector P
96: **end if**

Assuming $\alpha(k) = \alpha_j$ is the action chosen. The agent asks the agent j for help and receives the reinforcement β_k. In this $\beta_k$is given the similarity score between the current document and the chosen class computed by the helping agent. Then, the agent updates $\hat{d_j}$:

$$S_j(k+1) = S_j(k) + 1 \tag{3}$$

$$\hat{d_j}(k+1) = (S_j(k) * \hat{d_j}(k) + \beta_k)/(S_j(k)+1) \tag{4}$$

If $\hat{d_i}(k) = max_{i=1}^{n} \hat{d_i}(k)$, then $E_i(k) = 1$; otherwise $E_i(k) = 0$. After that, update the p vector by

$$p_i(k+1) = p_i(k) + \lambda * (E_i(k) - p_i(k)) \tag{5}$$

where λ is a constant that controls the learning speed and convergence.

The objective of using the action probability vector is to provide a chance for the agent to try all actions probabilistically. If an action is attempted sufficiently often, the estimate of its reward (the corresponding element of $\hat{d}$ vector) will be sufficiently close to its true value if there is one. This implies that one element in the E vector will result in the optimal action. This, in turn, will result in the convergence of the P vector to E, and hence, the optimal action. The convergence property of the Pursuit Algorithm has been proven [18].

3.3 Learning Algorithm: Nearest Centroid Learning

The Pursuit Learning algorithm described above assumes that neighboring agents that have helped will also be help-

ful in the future. However, this assumption is hardly true. The documents to be classified are diverse–an agent that helped to classify one type of documents does not necessarily succeed in another. For agents to learn and adapt to this diversity, we also developed a content-sensitive learning algorithm called Nearest Centroid Learning. The hypothesis is that an agent will be able to help classify documents "similar" to previous documents that the agent has helped with.

In this learning model, each agent has an array of centroid vectors $C = [c_1, ..., c_n]^N$, each of which corresponds to a neighboring agent/classifier. Initially, the array contains null centroid vectors. An agent also has a collaboration probability vector $P = [p_1, ..., p_n]^N$, which has a number of probability values corresponding to these neighbors.

An exploration rate value r is assigned in each experiment and remains constant for all agents throughout that experiment. Initially, without any collaboration, each value in vector P is assigned the value 1/n (n is the total number of its neighbors) equally. When the agent fails to classify a document, it generates a random number between [0,1]. If the number is between [0,r], then the agent randomly chooses another agent for help. Otherwise, it asks the agent with the maximum probability p_m.

If the helping agent fails, then nothing happens at this moment. If it otherwise classifies the document successfully, the agent being helped updates the centroid corresponding to the helping agent by adding the document vector and re-computing the averages. Thus, after several iterations of successful collaborations, there will be more non-null centroids of the documents that the corresponding agents have helped with. When the agent fails to classify a document again, it computes the P vector using the following steps with the exploration rate r and makes a prediction:

91: **for** $i = 0; i < n; i++$ **do**
92: $P[i] \leftarrow CosineSimilarity(doc, c[i])$ {*doc* is the current document vector}
93: **end for**
94: $rnd \leftarrow Randomize(1.0)$
95: **if** $rnd < r$ **then** {The agent "explores"}
96: Select an action randomly regardless of vector P
97: **else** {The agent predicts based on learned knowledge}
98: Select an action that has the maximum value in vector P
99: **end if**

4. EXPERIMENTAL DESIGN

4.1 The Reuters News Dataset

We used the corrected Reuters Corpus Volumes 1 (RCV1-v2) [12], a pre-labeled news collection made available by Reuters, Ltd., for the document classification tasks. First, we extracted the documents with single labels (i.e. each document belongs to one class). This process resulted in 8,894 documents, which were divided into a training set of 6,394 documents and a test set of 2,500 documents. After removing stop words and numbers, we stemmed the terms and weighted them using TF*IDF. Then, the terms were selected using a threshold on IDF values. 4,084 unique terms were preserved.

The 6,394 documents were used for offline training of the classifiers. This produced 37 document centroids for class vector representation. In the centralized approach, we used all the class vectors in one classifier. In the distributed environment, they were distributed among 37 agents equally, i.e. each agent has one unique class vector. Then we used the remaining 2,500 documents to run classification tests.

4.2 Evaluation Methodologies

Evaluating the effectiveness of learning requires proper baselines for comparison. There are two extreme cases of the agent society topology: a) the centralized approach with global knowledge and b) a distributed agent environment without any collaboration. The former defines an upper bound while the latter serves as a lower bound baseline. Hypothetically, an effective learning algorithm will help agents collaborate with others properly and achieve better results, approaching the upper bound. The following describes variables that will be controlled and/or measured in the experiments.

In our experiments, we will use the number of agents to control the knowledge distribution in the agent society, employ the two learning algorithms for agent collaboration, and measure their effectiveness. We will also study the impact of the following learning parameters: 1) the Exploration Rate r that controls the probability an agent makes decisions based on its learned memory or in a random way, and 2) the Maximum Collaboration Range g that denotes how deep a collaboration request can be forwarded when none of the helper agents has succeeded to help (see Figure 1).

To measure the experimental results, we look at the following aspects: effectiveness, efficiency, and the learning dynamics. While efficiency can be evaluated based on the total classification time, the learning dynamics will be analyzed by studying classification quality over time. For measuring classification quality or effectiveness, we will use Precision, Recall, and F measure as described below.

Table 1: A contingency table

	Expert Says Yes	Expert Says No
System Says Yes	a	b
System Says No	c	d

Lewis [11] used a contingency table (Table 1) to summarize the relationship between the system classifications and the expert judgements for a set of classification decisions. Then Precison, Recall, and F can be computed for each class given:

$$P = a/(a+b) \tag{6}$$

$$R = a/(a+c) \tag{7}$$

$$F_1 = 2 * P * R/(P+R) \tag{8}$$

Our document collection has 37 class labels. However, the standard deviation of #labeled documents across the classes is extremely high because of rare classes. In the training set, the maximum has 2,840 documents while 6 classes do not have documents. The average number of documents in each class is 173 with a standard deviation of 521. Given that the micro-averaging is more influenced by classification performance on common classes while the macro-averaging is influenced by classification performance on rare classes [29], we will use the micro-F to present our results.

4.3 Software and Hardware Setup

For multi-agent classification experiments, we have developed a software framework called Multi-Agent Collaboration for Classification of Information (MACCI). The second version MACCI II takes advantage of an agent platform named Cougaar Agent Architecture [6] and a machine learning framework called Weka [27]. MACCI II integrates the two major software packages (both in Java) to facilitate research experiments for modeling information retrieval based on agent collaboration and learning.

Generally, each agent has a classifier plug-in to take care of the classification tasks and can be configured to load a variety of classification algorithms. A collaboration plug-in includes several modules for the following purposes: to ask for help when an agent fails to classify a document, to respond to a help request when it is needed, and to acknowledge and thus learn from a helping agent when it gets helped. Each agent has a blackboard for relaying messages to each other. The collaboration plugin is also configurable so that different collaboration/learning algorithms can be integrated.

The experiments were conducted on one Redhat Linux AS 4 server with dual Intel Xeon 2.8 GHz CPUs and 3.5 GB RAM. 2GB memory was reserved for the experiments. The Java Runtime Environment version for this was 1.5.0_03.

5. EXPERIMENTAL RESULTS

With the MACCI framework, we conducted the following experiments on the RCV-v2 document collection. Firstly, we used a single agent with all the 37 class vectors to do classification. This simulated a traditional/centralized classifier and produced the upperbound baseline result. Then we increased the number of agents to $[2, 4, 8, 18, 37]$ and distributed the class vectors evenly among the agents–some agents might have one more class if the class vectors could not be divided equally. The larger the number of agents, the more distributed the knowledge of the agent community. On this stage, we did not introduce collaboration to the agent community so that we could isolate the impact of knowledge distribution.

On the second stage, we continued with the 37-agent community (i.e. the most distributed enviroment) and employed the learning algorithms (i.e. the Pursuit Learning and the Nearest-Centroid Learning) for agent collaboration. With each algorithm, we tried different g, r values and recorded both effectiveness and efficiency. We present the results below.

5.1 Classification Effectiveness

5.1.1 Effectiveness baselines

Experiments of distributed classification without collaboration produced the classification effectiveness (quality) baselines plotted below:

Figure 2 depicts the classification effectiveness vs. the number of agents, each of which divides the global knowledge equally and does not collaborate with others at all. Figure 3 draws the same picture on log/log coordinates to show tendencies more clearly. Note that when the number of agents is one, the situation is identical to a centralized approach where global knowledge is available in one place. As we expected, when the number of agents increases, Precision, Recall, and F decrease because knowledge (i.e. class vectors) becomes more distributed / sparse. From the log/log coordinates, it is very noticeable that Recall and F measure decrease dramatically while Precision looks a little more stable. The stabilization of Precision is understandable given the classification threshold and the definition of Precision (see Equation 6). When knowledge became more limited (distributed), the agents produced fewer correct answers, i.e. a smaller a in Table 1. However, due to the threshold, they also made fewer positive decisions, i.e. a smaller $a + b$ in Table 1.

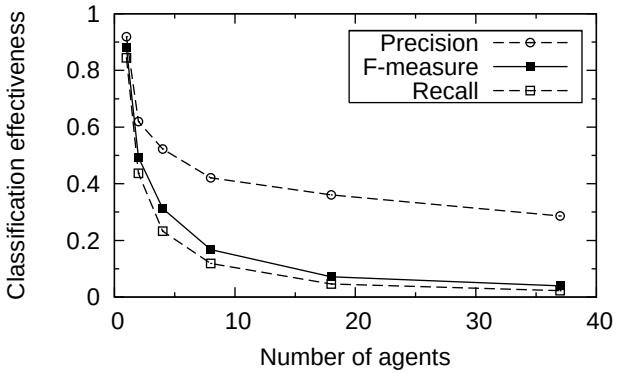

Figure 2: Classification effectiveness baseline

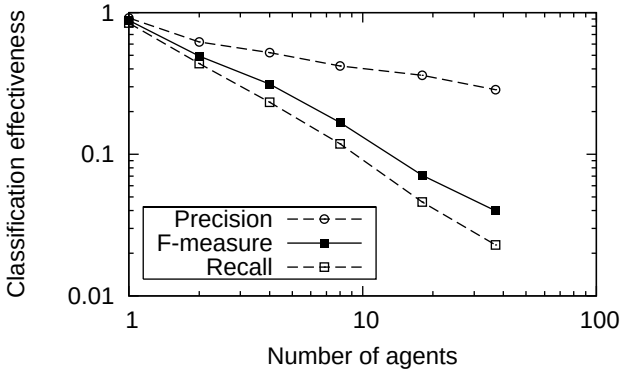

Figure 3: Effectiveness baseline (log/log)

These results will serve as two baselines for measuring classification effectiveness. The result achieved by one agent, i.e. the centralized approach, represents the best possible classification effectiveness. The results produced by distributed agents without collaboration produce a lower-bound baseline given the assumption that collaboration and learning can improve the results to some degree. Table 2 presents the baselines and some results achieved by the Pursuit Learning (PL) and the Nearest Centroid Learning (NCL). To study the effectiveness of learning in the experiments that follow, our focus is on the F measure that combines both Precision and Recall.

5.1.2 Effectiveness of learning

Classification effectiveness vs. the max collaboration range

To test the effectiveness of learning, we first ran the experiments for the two learning algorithms with a fixed exploration rate $r = 0.1$ and different max collaboration ranges.

Table 2: Baselines and some results for 37 agents

Method	g	r	R	P	F	Time (s)
Centralized	/	/	0.845	0.919	0.880	839
Non-collab	/	/	0.023	0.286	0.040	841
PL	8	.05	0.544	0.806	0.645	1022
NCL	8	.0	0.596	0.771	0.670	4205
PL	32	.1	0.681	0.753	0.715	1142
NCL	32	.1	0.558	0.719	0.628	7444

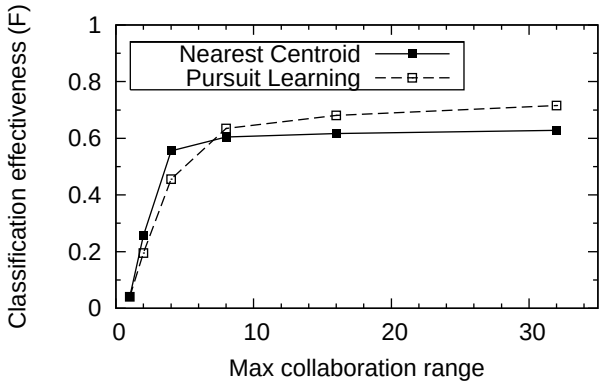

Figure 4: Classification effectiveness vs. max collaboration range ($\#agents = 37, r = 0.1$ **while** $g \in [2^0, 2^1, .., 2^5]$)

The classification effectiveness results in terms of F measure are shown in Figure 4.

Note that when the max collaboration range is one, the agents do NOT collaborate at all, representing the lower-bound baseline. As shown in 4, both the Pursuit Learning and the Nearest Centroid Learning improve classification quality when the max collaboration range increases. This indicates that a larger max range provides more opportunities for agents to collaborate with each other and increases the probability of classifying more documents correctly. The content-based Nearest Centroid learning achieved higher effectiveness than the Pursuit Learning when the max range remained small ($g \leq 4$). However, when $g \geq 8$, the Pursuit Learning was superior compared to the Nearest Centroid Learning. This suggests that the content-based learning was more accurate in predicting a helping agent within a small collaboration range but seemed "reluctant" to discover more when it was given more opportunities to collaborate. We will take a closer look at this in later.

Classification effectiveness vs. the exploration rate

To measure the impact of the exploration rate on the learning effectiveness, we ran the experiments for the two algorithms with a fixed collaboration range ($g = 8$) and different exploration rates ($r \in [0, 0.01, 0.02, 0.05, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0]$). Note that when the exploration $r = 0$, predicting a helping agent is exclusively based on what an agent learned without random exploration. When $r = 1.0$, it is random.

As shown in Figure 5, there exists an optimal point/zone ($r \approx 0.05$) for the Pursuit Learning. Classification effectiveness (F) decreases on both sides of the optimal point. On the one hand, the introduction of the randomness enables the learning algorithm to explore unlearned/unfamiliar actions.

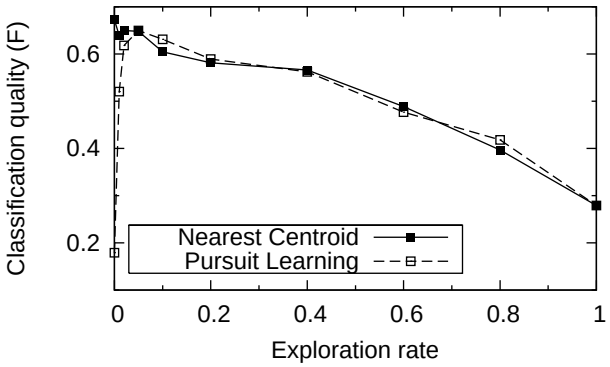

Figure 5: Classification effectiveness vs. exploration rate ($\#agents = 37, g = 8$ **while** $r \in [0, 0.01, 0.02, 0.05, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0]$)

On the other hand, the randomness also introduces noise to the classification tasks. For the Nearest Centroid Learning, the classification quality is the best when $r = 0$ and decreases when the exploration rate increases. The Nearest Centroid Learning actually explores even when $r = 0$. When an agent cannot find a relevant centroid for an incoming document, every element in the probability vector P is zero. In this case, the learner will choose an action randomly. For the Nearest Centroid learning, this randomness probably had been sufficient for exploration. Therefore, increasing its exploration rate did not help.

5.2 Learning Progression and Latency

5.2.1 Cumulative classification effectiveness over time

To study the effectiveness of a learning algorithm, close investigation of its learning dynamics is helpful. We used two approaches for this purpose. In the first approach, we looked at cumulative classification effectiveness over time.

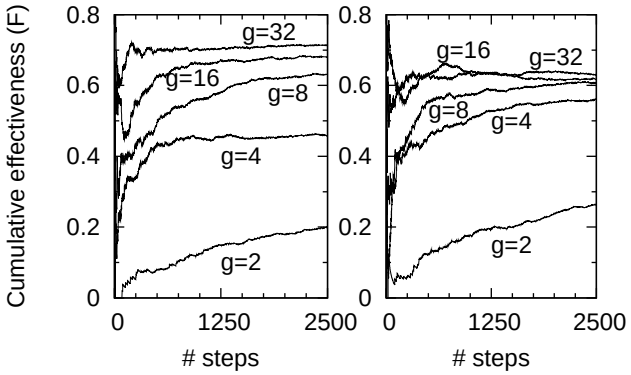

Figure 6: Cumulative classification effectiveness over time vs. max collaboration range
Left: Pursuit Learning; Right: Nearest Centroid Learning. $\#agents = 37, r = 0.1$ while $g \in [2, 4, 8, 16, 32]$.

Figure 6 shows clear trends of increasing cumulative classification effectiveness. Both the Pursuit Learning and the Nearest Centroid Learning stabilize or show tendencies to stabilize after 500 steps or so. When the max range is larger

(e.g. 16, 32), the cumulative classification effectiveness stabilizes more quickly and at a higher level. This is understandable as agents have more opportunities to collaborate and learn with a larger max collaboration range.

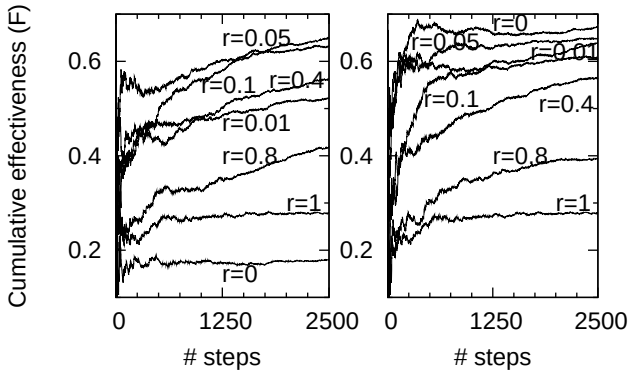

Figure 7: Cumulative classification effectiveness over time vs. exploration rate
Left: Pursuit Learning; Right: Nearest Centroid Learning. $\#agents = 37, g = 8$ while $r \in [0, 0.01, 0.05, 0.1, 0.4, 0.8, 1.0]$.

Figure 7 draws the learning curves with different exploration rates. Apparently, the learning curve with $r = 1$ (totally random) and the Pursuit Learning curve with $r = 0$ (no randomness) stabilized very quickly and were not able to achieve high effectiveness. When $r = 0$, the Pursuit Learning lost its capacity to explore and to update its memory after it had been initialized–once it received the first successful help from another, it more or less stuck to that helping agent. When $r = 1$, the learning algorithm was purely random and would never learn more for better prediction.

As was discussed, with $g = 8$, the Pursuit Learning had an optimal point for classification effectiveness where $r \approx 0.05$. On Figure 7, the Pursuit Learning (left) with $r = 0.05$ appears on the top and still shows a tendency of increasing at the end. This suggests the learning algorithm can achieve even better results with a larger document collection. A similar pattern is found with the best result of the Nearest Centroid Learning ($r = 0$).

5.2.2 *In-session classification effectiveness over time*

In the second analysis, the 2,500 documents were divided into 100 sessions, each of which had 25 documents (steps). Then we computed in-session classification effectiveness for those sessions and plotted a moving average every 10 sessions. We applied this method to the best results produced by the Pursuit Learning and the Nearest Centroid Learning respectively with a fixed max range $g = 8$ in Figure 8.

Again, Figure 8 shows that neither the Pursuit Learning nor the Nearest Centroid Learning converge. Both had the tendency to increase. Obviously, the collection of 2,500 test documents was not sufficiently big for the learning algorithms to converge in these experiments. Future research with a larger test collection will probably produce better results.

Yet as shown in Figure 6, when $g \in [16, 32]$ with a fixed $r = 0.1$, the cumulative classification effectiveness stabilized. To examine the convergence of the learning algorithms closely, we applied the method to results produced with $r = 0.1, g = 16$ (Figure 9).

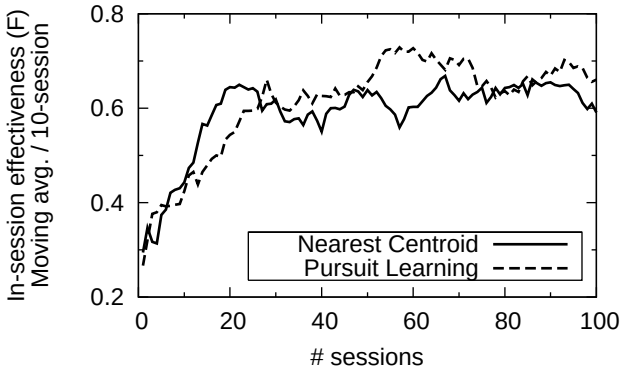

Figure 8: In-session classification effectiveness over time ($\#agents = 37, g = 8$)
Pursuit Learning, $r = 0.05$; Nearest Centroid, $r = 0$.

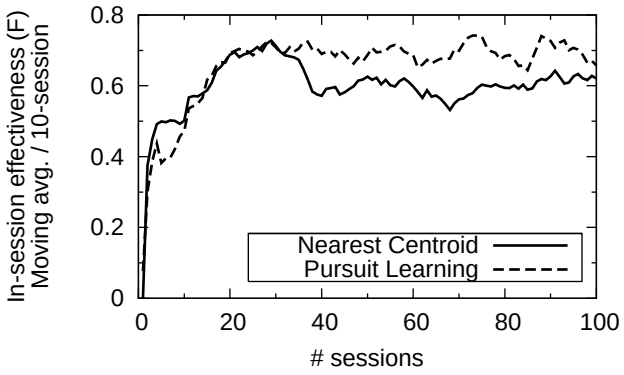

Figure 9: In-session classification effectiveness over time ($\#agents = 37, g = 16, r = 0.1$)

Figure 9 shows that the Pursuit Learning stabilizes very quickly. The Nearest Centroid Learning does not stabilize at the end, suggesting that further experimental investigation of this algorithm is needed.

5.3 Classification Efficiency

5.3.1 *Efficiency baselines*

Classification time was recorded in each experiment and analyzed. Experiments of distributed classification without collaboration produced the classification efficiency (time) baselines plotted in Figure 10.

All these baseline experiments were conducted on one server, meaning that the overall computing resource was fixed. When the number of agents increases, knowledge (class vectors) used for classification becomes more limited, thus consuming less computing resource for each classification task. On the other hand, each agent thread also consumes computing time even without running classification tasks. The tradeoffs of these two factors resulted in the curve shown in Figure 10.

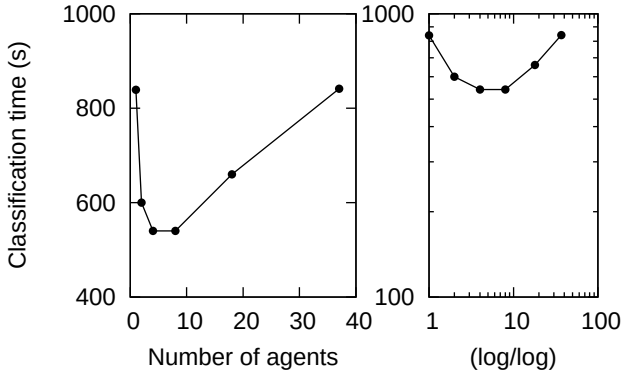

Figure 10: Classification efficiency baseline
Right: log/log coordinates.

5.3.2 *Efficiency vs. max collaboration range*

Figure 11 shows the overall classification time vs. the max collaboration range given a fixed exploration rate $r = 0.1$.

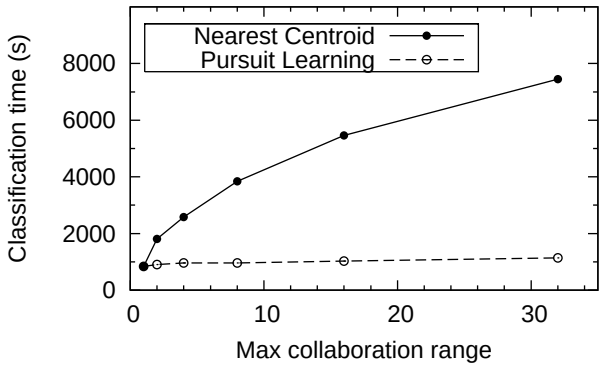

Figure 11: Classification efficiency vs. Max collaboration range ($\#agents = 37, r = 0.1$)

When the max range is one, there is no collaboration at all (the efficiency baseline). As was expected, when the max range increases, it takes longer to do classification. Classification time of the Nearest Centroid Learning increases much more quickly than the Pursuit Learning, indicating the content-based learning algorithm is more time consuming. The continuous increasing pattern of the Nearest Centroid Learning also suggests that the larger max range did trigger more collaborations. In other words, in a certain percentage of cases, the learning algorithm was not able to find an agent that could offer successful help within a few steps, i.e. a range smaller than the max range. Otherwise, the curve would have stabilized when the max range was sufficiently big (e.g. 32).

5.3.3 *Efficiency vs. exploration rate*

Figure 12 shows the overall classification time vs. the exploration rate given a fixed max collaboration range $g = 8$.

As shown in the Figure 12 (the zoom-in view), the Pursuit Learning achieved optimal efficiency when $r \in [0.1, 0.4]$. This zone also relates to the optimal zone in Figure 5 where the classification qualities are on the top. This is encouraging as it indicates that there does exist an optimal zone where high effectiveness and high efficiency meet.

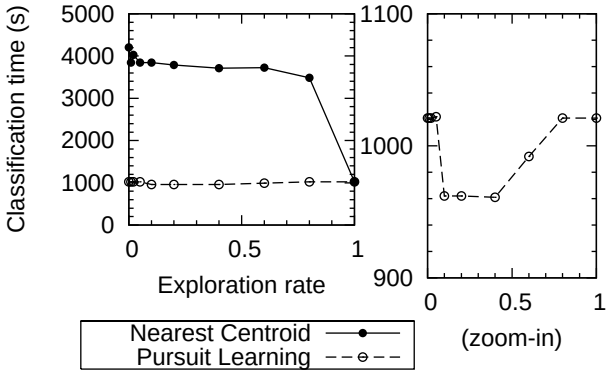

Figure 12: Classification efficiency vs. Exploration rate ($\#agents = 37, g = 8$)
Right: zoom-in of the Pursuit Learning curve.

5.3.4 *Efficiency vs. effectiveness*

To correlate efficiency (classification time) and effectiveness (F measure), we plotted all experimental results (Figure 13). Some of the Pursuit Learning points come very close to the centralized baseline, meaning the algorithm was not only effective but aslo efficient in some experiments. The Nearest Centroid Learning algorithm was effective but time consuming and requires further investigation.

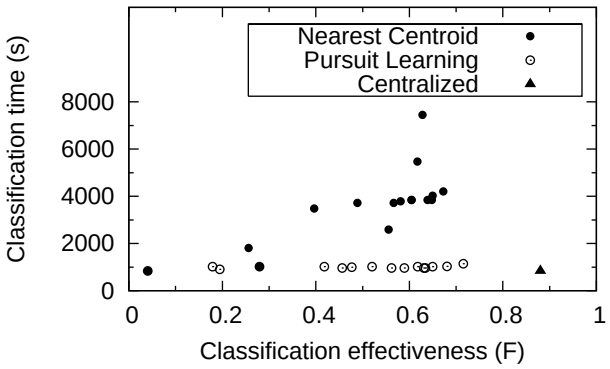

Figure 13: Classification efficiency vs. Effectiveness

6. SUMMARY AND FUTURE WORK

As the experimental results have shown, classification effectiveness of distributed agents without collaboration is lower than the centralized approach and decreases dramatically when knowledge becomes increasingly distributed. The two learning algorithms used in our experiments for collaboration can both overcome the knowledge distribution effectively and improve the classification quality to a level comparable to the centralized baseline.

As we expected, the Pursuit Learning was more efficient than the Nearest Centroid Learning given that the latter had to analyze a document content at each learning step. Surprisingly, the Pursuit Learning algorithm, although not

content-sensitive, achieved competitive classification effectiveness. Comparing the results with the centralized and the distributed baselines showed that there did exist an optimal zone for the Pursuit Learning where high efficiency and high effectiveness met.

The Pursuit Learning approach did not depend on document content. By acquiring knowledge through reinforcements based on collaborations this algorithm was able to construct/build paths for documents to find relevant classifiers effectively and efficiently. The Nearest Centroid Learning algorithm did not converge in several cases but produced some interesting and stable patterns. These results convinced us that taking into account document content should bring some advantages to the learning process. Hence, combining the advantages of the two learning approaches and conducting experiments on larger test collections may be a potentially useful direction for improving distributed IR systems.

In the current simulation model, every agent potentially knows about all neighboring agents and at least maintains a pointer to each other. This is manageable with a moderate number of agents but unrealistic when the number of agents scales up. In a huge agent community, agents have neither enough time nor capacity to interact and keep track of all the others. Future research will involve studying agent collaboration for document classification when each agent is given a limited number of neighbors in a larger scale agent society.

7. ACKNOWLEDGMENTS

Authors thank Kiduk Yang, Katy Börner, Larry Yaeger, Ketan Mane, Shashikant Penumarthy, and Gavin La Rowe for valuable discussions and feedback. We also appreciate constructive comments given by the anonymous reviewers of the JCDL 2007 conference. This project is partially supported by the NSF grant ENABLE #0333623.

8. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley Longman Publishing, 2004.

[2] K. Börner, L. Dall'Asta, W. Ke, and A. Vespignani. Studying the emerging global brain: Analyzing and visualizing the impact of co-authorship teams. *Complexity, special issue on Understanding Complex Systems*, 10(4):58–67, April 2005.

[3] J. Callan, F. Crestani, and M. Sanderson. Sigir 2003 workshop on distributed information retrieval. *SIGIR Forum*, 37(2):33–37, 2003.

[4] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, S. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Mining the link structure of the world wide web. *IEEE Computer*, 32(8):60–67, August 1999.

[5] Y. Fu, W. Ke, and J. Mostafa. Automated text classification using a multi-agent framework. In *JCDL '05: Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pages 157–158, New York, NY, USA, 2005. ACM Press.

[6] A. Helsinger, M. Thome, and T. Wright. Cougaar: A scalable, distributed multi-agent architecture. In *Proceedings of IEEE Conference on Systems, Man and Cybernetics 2004*, October 2004.

[7] W. Ke, Y. Fu, and J. Mostafa. Advanced information retrieval web services for digital libraries. *Library Collections, Acquisitions, and Technical Services*, 29(2):220–224, 2005.

[8] K. Knight. Mining online text. *Commun. ACM*, 42(11):58–61, 1999.

[9] R. R. Korfhage. *Information Storage and Retrieval*. Wiley Computer Pub, New York, 1997.

[10] D. D. Lewis. *Text representation for intelligent text retrieval: a classification-oriented view*, pages 179–197. Hillsdale, NJ: Lawrence Erlbaum, 1992.

[11] D. D. Lewis. Evaluating and Optimizing Autonomous Text Classification Systems. In E. A. Fox, P. Ingwersen, and R. Fidel, editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–254, Seattle, Washington, 1995. ACM Press.

[12] D. D. Lewis, Y. Yang, and T. Rose. A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[13] F. Menczer and R. K. Belew. Adaptive information agents in distributed textual environments. In *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, pages 157–164, New York, NY, USA, 1998. ACM Press.

[14] P. J. Modi and W.-M. Shen. Collaborative multiagent learning for classification tasks. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 37–38, New York, NY, USA, 2001. ACM Press.

[15] J. Mostafa and W. Lam. Automatic classification using supervised learning in a medical document filtering application. *Information Processing & Management*, 36(3):415–444, 2000.

[16] S. Mukhopadhyay, S. Peng, R. Raje, J. Mostafa, and M. Palakal. Distributed multi-agent information filtering - a comparative study. *Journal of the American Society for Information Science*, 56(8):834–842, 2005.

[17] S. Mukhopadhyay, S. Peng, R. Raje, M. Palakal, and J. Mostafa. Multi-agent information classification using dynamic acquaintance lists. *Journal of the American Society for Information Science and Technology*, 54(10):966–975, 2003.

[18] S. Mukhopadhyay and M. Thathachar. Associative learning of boolean functions. *IEEE Transactions on Systems, Man and Cybernetics*, 19:1008–1015, 1989.

[19] S. Peng, S. Mukhopadhyay, R. Raje, and M. Palakal. A comparison between single-agent and multi-agent classification of documents. In *IPDPS '01: Proceedings of the 10th Heterogeneous Computing Workshop â"" HCW 2001 (Workshop 1)*, page 20090.2, Washington, DC, USA, 2001. IEEE Computer Society.

[20] F. B. Pereira and E. Costa. The influence of learning in the behavior of information retrieval adaptive agents. In *Proceedings of the Symposium of Applied Computing*, March 2002.

[21] V. G. Red'ko. Problem of the global brain and multi-agent modeling. *ICAIS*, 00:279–282, 2002.

[22] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.

[23] F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.

[24] O. E. Taulbee. Invited papers–1: classification in information storage and retrieval. In *Proceedings of the 1965 20th national conference*, pages 119–137, New York, NY, USA, 1965. ACM Press. Chairman-R. W. House.

[25] V. S. Vladimir Gorodetsky, Oleg Karsaeyv. Multi-agent technology for distributed data mining and classification. In *IEEE/WIC International Conference on Intelligent Agent Technology (IAT'03)*, page 438, Los Alamitos, CA, USA, October 2003. IEEE Computer Society.

[26] M. Wimmer and R. Traunmuller. Trends in electronic government: Managing distributed knowledge. *DEXA*, 00:340–345, 2000.

[27] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.

[28] K. Yang. *Combining Text-, Link-, and Classification-based Retrieval Methods to Enhance Information Discovery on the Web*. PhD thesis, University of North Carolina at Chapel Hill, 2002.

[29] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49, New York, NY, USA, 1999. ACM Press.

[30] E. S. Yu, P. C. Koo, and E. D. Liddy. Evolving intelligent text-based agents. In *AGENTS '00: Proceedings of the fourth international conference on autonomous agents*, pages 388–395, New York, NY, USA, 2000. ACM Press.