TopicSearch—Personalized Web Clustering Engine Using Semantic Query Expansion, Memetic Algorithms and Intelligent Agents

Carlos Cobos, Martha Mendoza, Elizabeth León, Milos Manic, and Enrique Herrera-Viedma

***Abstract*—As resources become more and more available on the Web, so the difficulties associated with finding the desired information increase. Intelligent agents can assist users in this task since they can search, filter and organize information on behalf of their users. Web document clustering techniques can also help users to find pages that meet their information requirements. This paper presents a personalized web document clustering called TopicSearch. TopicSearch introduces a novel inverse document frequency function to improve the query expansion process, a new memetic algorithm for web document clustering, and frequent phrases approach for defining cluster labels. Each user query is handled by an agent who coordinates several tasks including query expansion, search results acquisition, preprocessing of search results, cluster construction and labeling, and visualization. These tasks are performed by specialized agents whose execution can be parallelized in certain instances. The model was successfully tested on fifty DMOZ datasets. The results demonstrated improved precision and recall over traditional algorithms (k-means, Bisecting k-means, STC y Lingo). In addition, the presented model was evaluated by a group of twenty users with 90% being in favor of the model.**

***Index Terms*—Web document clustering, intelligent agents, query expansion, WordNet, memetic algorithms, user profile.**

I. Introduction

In recent years, web document clustering has become a very interesting research area among academic and scientific communities involved in information retrieval (IR) and web search [1]. Web document clustering systems seek to increase the coverage (amount) of documents presented for the user to review, while reducing the time spent in reviewing documents [2]. In IR, these web document clustering systems are called web clustering engines. Among the most prominent ones are Carrot (www.carrot2.org), SnakeT (snaket.di.unipi.it), Yippy (yippy.com, originally named as Vivisimo and then as Clusty), iBoogie (www.iboogie.com), and KeySRC (keysrc.fub.it) [3]. Such systems usually consist of four main components: search results acquisition, preprocessing of input, cluster construction and labeling, and visualization of resulting clusters [1] (see Fig 1).

The **search results acquisition** component begins with a query defined by the user. Based on this query, a document search is conducted in diverse data sources, in this case in the traditional web search engines such as Google, Yahoo! and Bing. In general, web clustering engines work as meta search engines and collect between 50 to 200 results from traditional search engines. These results contain as a minimum a URL, a snippet and a title [1].

The **preprocessing** of search results comes next. This component converts each of the search results (as snippets) into a sequence of words, phrases, strings or general attributes or characteristics, which are then used by the clustering algorithm. There are a number of tasks performed on the search results, including: removing special characters and accents, the conversion of the string to lowercase, removing stop words, stemming of the words and the control of terms or concepts allowed by a vocabulary [1].

Once the preprocessing is finished, **cluster construction and labeling** is begun. This stage makes use of three types of algorithm [1]: data-centric, description-aware and description-centric. Each of these builds clusters of documents and assigns a label to the groups.

Data-centric algorithms are the algorithms traditionally used for data clustering (partitional, hierarchical, density-based, etc.) [1, 4-10]. They look for a solution in data clustering, but lack in their capabilities presentation of the labels and in providing explanations of the groups obtained. These algorithms address the problem of web document clustering as merely another data clustering problem.

Description-aware algorithms put more emphasis on one specific feature of the clustering process. For example, they might put a priority on the quality of the labeling of groups and as such achieve results that are more easily interpreted by the user. The quality of these algorithms however deteriorates during the cluster creation process. An example of this type of algorithm is Suffix Tree Clustering (STC) [8], which incrementally creates labels easily understood by users, based on common phrases that appear in the documents.

Manuscript received on March 13, 2013; accepted for publication on May 23, 2013.

Carlos Cobos and Martha Mendoza are with the University of Cauca, Colombia (e-mail: {ccobos,mmendoza}@unicauca.edu.co).

Elizabeth León is with the Universidad Nacional de Colombia, Colombia (e-mail: eleonguz@unal.edu.co).

Milos Manic is with the University of Idaho at Idaho Falls, USA (e-mail: misko@uidaho.edu)

Enrique Herrera-Viedma is with University of Granada, Spain (e-mail: viedma@decsai.ugr.es)

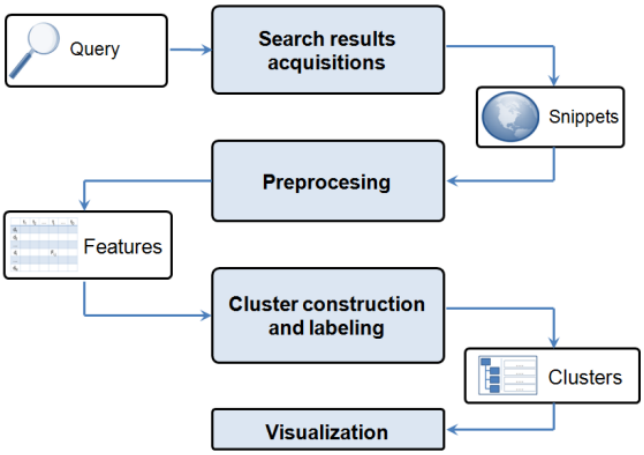

Fig 1.The components of a web clustering engine (adapted from [1])

Description-centric algorithms [1, 7, 11-15] are designed specifically for web document clustering, seeking a balance between the quality of clusters and the description (labeling) of clusters. An example of such algorithms is Lingo [11] (implemented by www.carrot2.org), which makes use of Singular Value Decomposition (SVD) to find the best relationships between terms, but groups the documents based on the most frequent phrases in the document collection.

Finally, in the **visualization** step, the system displays the results to the user in folders organized hierarchically. Each folder seeks to have a label or title that represents well the documents it contains and that is easily identified by the user. As such, the user simply scans the folders that are actually related to their specific needs. The presentation folder tree has been adopted by various systems such as Carrot2, Yippy, SnakeT, and KeySRC, because the folder metaphor is already familiar to computer users. Other systems such as Grokker and Kart004 use a different display scheme based on graphs [1].

In order to obtain satisfactory results in web document clustering, the algorithms must meet the following specific requirements [1, 8]: Automatically define the number of clusters that are going to be created; generate relevant clusters for the user and assign these documents to appropriate clusters; define labels or names for the clusters that are easily understood by users; handle overlapping clusters (the document can belong to more than one cluster); reduce the high dimensionality of document collections; handle the processing time i.e. less than or equal to 2.0 seconds; and handle the noise frequently found in documents.

Another important aspect of web document clustering algorithms is the document representation model. The most widely used models are [16]: Vector space model [2, 4], Latent Semantic Indexing (LSI) [2, 11], Ontology-based model [7, 17], N-gram [8], Phrase-based model [8], and Frequent Word (Term) Sets model [7, 18].

In Vector space model (VSM), the documents are designed as bags of words. Document collection is represented by a matrix of D-terms by N-documents. This matrix is commonly called Term by Document Matrix (TDM). In TDM, each document is represented by a vector of normalized frequency term by document inverse frequency for that term, in what is known as the TF-IDF value. In VSM, the cosine similarity is used for measuring the degree of similarity between two documents or between a document and the user's query. In VSM as in most of the representation models, a process of stop word removal and stemming [2] should be done before re-presenting the document. Stop word removal refers to the removal of very common words (like articles and prepositions, so can yield over 40% reduction on TDM matrix dimensionality), while stemming refers to the reduction of words to their canonical stem or root form.

The two predominant problems with existing web clustering are inconsistencies in cluster content and inconsistencies in cluster description [1]. The first problem refers to the content of a cluster that does not always correspond to the label. Also, the navigation through the cluster hierarchies does not necessarily lead to more specific results. The second problem refers to the need for more expressive descriptions of the clusters (cluster labels are confusing).This is the main motivation of the present work, in which a personalized web clustering engine modeled by agents is put forward. This model is developed to work on-line and off-line, which means that users can define the processing time (for example, it can be fixed at a value of less than two seconds to work on-line) of agents in the entire process of search, clustering and visualization. To the best of our knowledge, this research is the first to integrate synergistically web document clustering, the semantic query expansion process (based on WordNet and user profile), and memetic algorithms through a model of intelligent agents.

The remainder of the paper is organized as follows. Section 2 presents related work. Section 3 presents a personalized web document clustering model, i.e. the query expansion process, the clustering and labeling algorithm, and the user profile. Section 4 shows the experimental results. Finally, some concluding remarks and suggestions for future work are presented.

II. Related Work

In general, clustering algorithms can be classified into [19]: hierarchical, partitional, density-based, grid-based, and model-based algorithms, among others. The algorithms most commonly used for web document clustering have been the hierarchical and the partitional [4]. The hierarchical algorithms generate a dendogram or a tree of groups. This tree starts from a similarity measure, among which are: single link, complete link and average link. In relation to web document clustering, the hierarchical algorithm that brings the best results in accuracy is called UPGMA (Unweighted Pair-Group Method using Arithmetic averages) [5]. UPGMA was

devised in 1990 [7] and is based on the vector space model, using an average link based on the clusters cosine similarity divided by the size of the two clusters that are being evaluated. UPGMA has the disadvantage of having a time complexity of $O(n^3)$ and being static in the process of assigning documents to clusters.

In partitional clustering, the algorithms perform an initial division of the data in the clusters and then move the objects from one cluster to another based on the optimization of a predefined criterion or objective function [19]. The most representative algorithms using this technique are: K-means, K-medoids, and Expectation Maximization. The K-means algorithm is the most popular because it is easy to implement and its time complexity is $O(n)$, where n is the number of patterns or records, but it has serious disadvantages: it is sensitive to outliers, it is sensitive to the selection of the initial centroids, it requires a prior definition of the number of clusters, and the obtained clusters are only hyper spherical in shape [8]. In 2000, a Bisecting K-means [4, 7] algorithm was devised. This algorithm combines the strengths of the hierarchical and partitional methods reporting better results concerning the accuracy and the efficiency of the UPGMA and the K-means algorithms.

The first algorithm to take the approach based on frequent phrases shared by documents in the collection was put forward in 1998 and called Suffix Tree Clustering (STC) [7, 8]. Later in 2001, the SHOC (Semantic, Hierarchical, Online Clustering) algorithm was introduced [12]. SHOC improves STC and is based on LSI and frequent phrases. Next in 2003, the Lingo algorithm [11, 20] was devised. This algorithm is used by the Carrot2 web searcher and it is based on complete phrases and LSI with Singular Value Decomposition (SVD). Lingo is an improvement of SHOC and STC and (unlike most algorithms), tries first to discover descriptive names for the clusters and only then organizes the documents into appropriate clusters.

NMF (also in 2003) is another example of these algorithms, it is based on the non-negative matrix factorization of the term-document matrix of the given document corpus was made available [21]. This algorithm surpasses the LSI and the spectral clustering methods in document clustering accuracy but does not care about cluster labels.

Another approach was proposed by the Pairwise Constraints guided Non-negative Matrix Factorization (PCNMF) algorithm [22] in 2007. This algorithm transforms the document clustering problem from an un-supervised problem to a semi-supervised problem using must-link and cannot-link relations between documents. In 2007, the Dynamic SVD clustering (DSC) [14] algorithm was made available. This algorithm uses SVD and minimum spanning tree (MST). This algorithm has better performance than Lingo. Finally, in 2008, the CFWS (Clustering based on Frequent Word Sequences) and the CFWMS (Clustering based on Frequent Word Meaning Sequences) [7] algorithms were proposed. These algorithms represent text documents as frequent word sequences and frequent concept sequences (based on WordNet), respectively and they are mostly used in text clustering.

In relation to a frequent word sets model for web document clustering, in 2002, FTC (Frequent Term-Based Text Clustering) and HFTC (Hierarchical Frequent Term-Based Text Clustering) algorithms became available [15]. These algorithms use combinations of frequent words (association rules approach) shared in the documents to measure their proximity in the text clustering process.

Then in 2003, FIHC (Frequent Item set-based Hierarchical Clustering) was introduced [13] which measures the cohesion of a cluster using frequent word sets, so that the documents in the same cluster share more of the frequent word sets than those in other groups. These algorithms provide accuracy similar to that reported for Bisection K-means, with the advantage that they assign descriptive labels to associate clusters.

Finally, looking at partitional clustering from an evolutionary approach: in 2007, three hybridization methods between the Harmony Search (HS) [23] and the K-means algorithms [24] were compared. These were: Sequential hybridization method, interleaved hybridization method and the hybridization of K-means as a step of HS. As a general result, the last method was the best choice of the three. Later, in 2008 [9, 23, 25], based on the Markov Chains theory the researchers demonstrated that the last algorithm converges to the global optimum.

Next, in 2009, a Self-Organized Genetic [17] algorithm was devised for text clustering based on the WordNet ontology. In this algorithm, a modified LSI model was also presented, which appropriately gathers the associated semantic similarities. This algorithm outperforms the standard genetic algorithm [26] and the K-means algorithm for web document clustering in similar environments. In 2010, two new algorithms were put forward. The first one, called IGBHSK [27] was based on global-best harmony search, k-means and frequent term sets. The second one, called WDC-NMA [28] was based on memetic algorithms with niching techniques. These two researches outperform obtained results with Lingo (Carrot2) over few datasets.

III. The Proposed Model: TopicSearch

TopicSearch is a personalized web clustering engine based on semantic query expansion, user profile, and memetic algorithms through a model of intelligent agents. In the proposed model, the web document clustering problem was transformed from an on-line scenario to an on-line and off-line scenario. With this change, users can execute queries with instant response and users can send a query and see results later. In both scenarios, the results are very promising, but quality of results increases when the clustering algorithm is executed more time.

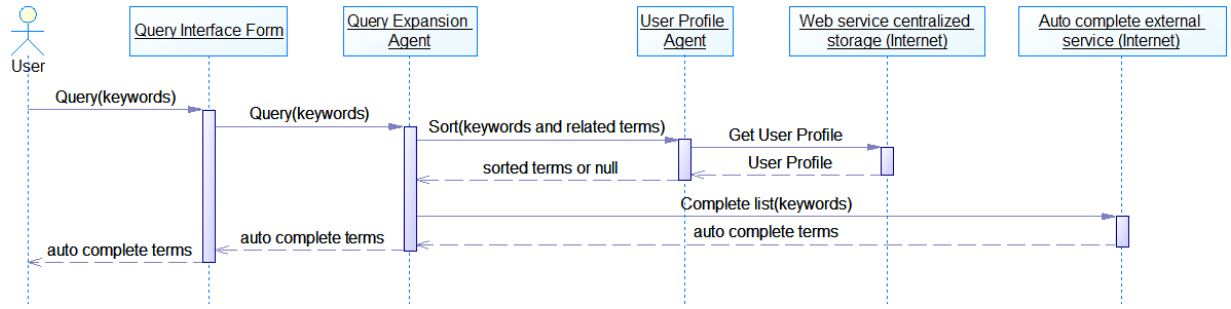

Fig 2. Agents in the Query Expansion Process

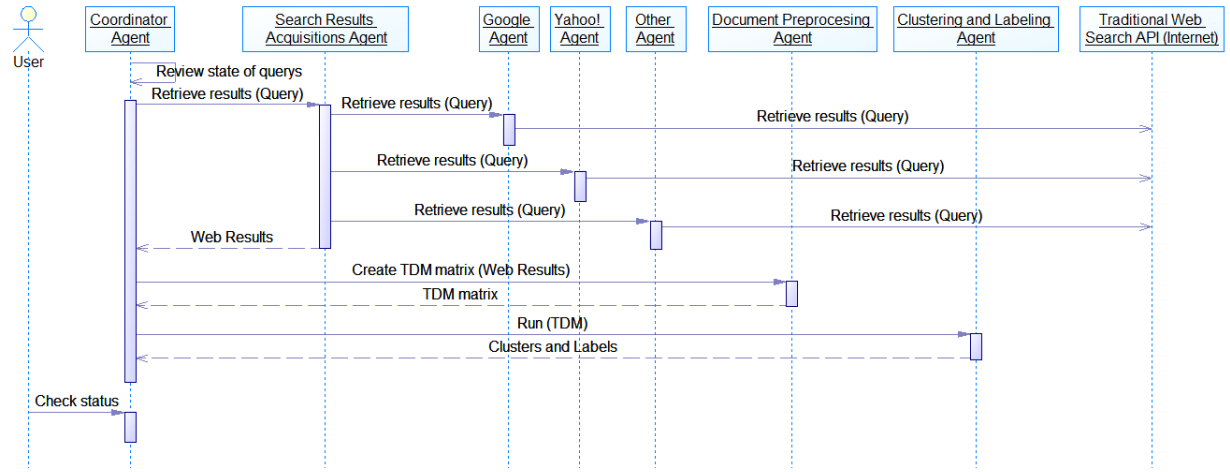

Fig 3. Agents in the Search and Clustering Process

The main actor in TopicSearch is the user. The user can execute multiple queries at the same time. Each query is handled by a group of agents. To present the model, the entire search process was organized into three sub processes: query expansion, search and clustering, and the visualization of results.

Query Expansion: When the user is typing the query it is supported by an interface agent called ***Query Expansion Agent***, which is responsible for displaying a list of terms that help the user to complete the query.

This agent uses the ***User Profile Agent*** which is responsible for finding through a web service the user profile data—in this case, a set of terms with its relevance and correlation among those terms. If the User Profile Agent has no information from the user, the Query Expansion Agent uses an external service of auto-complete, for example, the auto-complete Google service (Fig 2).

Search and Clustering: When the user starts the process of searching, a ***Coordinator Agent*** is activated for each specific query. This agent activates a ***Search Results Acquisitions Agent*** in order to retrieve the results of traditional web search engines like Google, Yahoo!, Bing, etc. At this point the Search Results Acquisitions Agent generates many agents as external web search services registered in the model, thereby achieving a parallel job which reduces the processing time at this stage of the process. In Fig 3 these agents are called ***Google Agent***, ***Yahoo! Agent*** and ***Other Agent***. When the results acquisition process ends, Coordinator Agent activates a ***Document Preprocessing Agent*** in charge of creating a matrix of terms by documents or TDM matrix. After the construction of the TDM matrix, Coordinator Agent activates the ***Clustering and Labeling Agent***, which is responsible for creating clusters of documents based on a memetic algorithm called Agents-WDC and assigns labels to the clusters based on a frequent phrases approach.

As a result, the model obtains a Clustered Documents and Cluster Labels which can be viewed by the user at any time (Fig 3).

Visualization of results: The user visualizes a form with a list of queries that he/she had previously registered in the system. Each query shows a status (Started, Completed, in-Evolution). When the Coordinator Agent is in the process of acquisition or pre-processing of results the query is in the Initiated state and cannot be stopped. When running the Clustering and Labeling Agent, the status of the query is in-Evolution and the process can be stopped. In this case the system generates the cluster labels of the best result found so far and goes on to state Completed. Finally, the completed state occurs when the Coordinator Agent ends the process for the query.

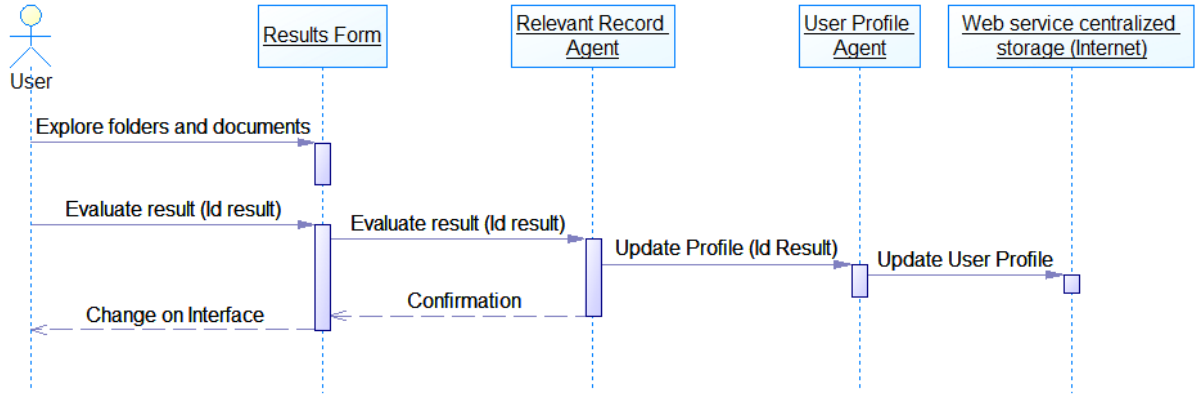

Fig 4. Agents in the Visualization of Results Process

When the user selects the results of a query, the system displays a form (interface) divided into two parts, the left side with a list of cluster labels and the right side with the list of documents belonging to each cluster label. When the user marks a document as relevant or not relevant, the ***Relevant Record Agent*** processes the document terms and through the ***User Profile Agent*** updates the user profile in the web service which centralizes the storage of users (see Fig 4).In this way, future queries can enjoy a more relevant search expansion process based on an updated profile.

The ***Web Service: Centralized Storage*** component allows users to log in from different computers and makes updated user profile information always available. Moreover, with the Windows Client Application (or Smart Client Application), the system takes advantage of the processing capacity of personal computers to reduce the workload of the centralized application server.

On the other hand, the deployment (installation and updates) for smart client applications is becoming increasingly easy to do. Next, we present a detailed description of different components of the model.

A. Query Expansion

In VSM, it has been shown that the process of query expansion improves the relevance (as measured by the accuracy) of the results delivered to users [2, 29, 30]. The expansion of the query in a web search system is usually made from one of two perspectives: user relevance feedback (URF) or automatic relevance feedback (ARF) [2, 29, 30]. URF requires the user to mark documents as relevant or not relevant.

The terms in these marked documents that the system has found to be relevant or not are added to or removed from each of the new user queries [2, 29, 30]. Rocchio proposes formula (1) to generate the expanded query, where q is the query typed by the user initially, R is a set of relevant documents, R′ is a set of non-relevant documents, α, β and γ are tuning constants for the model and q is the expanded query [2, 29, 30]:

$$q_e = \propto * q + \frac{\beta}{|R|}\sum_{d \in R} d - \frac{\gamma}{|R'|}\sum_{d \in R'} d. \quad (1)$$

In contrast, ARF (also known as pseudo feedback) expands the queries automatically based on two methods: considering global documents and considering partial documents [2, 29, 30].

In the global document-based methods, all documents in the collection are analyzed and relationships established between terms (words). As such, these methods are typically performed based on thesauri.

The disadvantage of these methods is that they need all the documents. In addition, the process of updating the thesaurus can be expensive and complex [2, 29, 30].

In the methods based on partial documents, the query is originally sent to the search engine. With the results delivered, a group of documents is selected (the first results, the most relevant) and with these the query is reformulated (Rocchio's formula with $\gamma = 0$) and re-sent to the search engine. The results of the second (or expanded) search are those which are actually presented to the user [2, 29, 30].

Both expansion models have some problems: for example one assumes that the user is always going to mark documents as relevant or not and the other assumes that the first results from the original query are all relevant [2, 29, 30].

Carpineto *et al*. [1] presented the need for giving more importance to the query expansion process in web clustering engines. TopicSearch offers a query expansion process that gives greater importance to the semantic similarity between terms (words), but leaves option for users to feedback into the model which documents are relevant, and which are not.

TopicSearch starts the search process with a user query (based on key words.) This query is expanded explicitly with the help of the user, through an auto-complete option. This option is based on a dynamic, drop-down list of terms that are displayed in a similar way to those of Google.

The auto-complete option is generated based on the list of terms that have been relevant to the user in earlier queries.

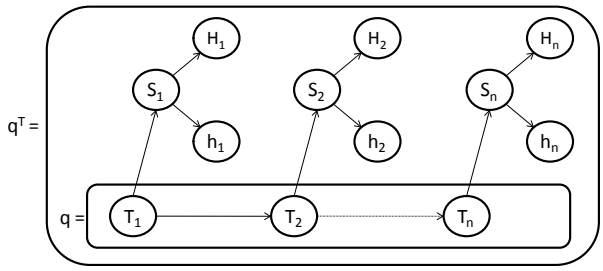

Fig 5. Expanded query for each term: synonyms (S), hypernyms (H) and hyponyms (h).

The process involves three steps described as follows: (1) pre-processing and semantic relationship, (2) terms listed in the profile and (3) using an external service.

1. **Pre-processing and semantic relationship**: It first takes the user query and removes special characters, converting each term to lower case and eliminating stop words. Then, it finds the most common synonyms (S, set of terms or synset in different languages that are used to represent the same concept), hypernyms (H, set of terms in the next level above in the hierarchy of the ontology, generalizations of the concept) and hyponyms (h, set of terms in the next level below in the hierarchy of the ontology, specializations of the concept) of the terms that the user has typed, based on WordNet (see Fig 5). In this research—as in WordNet—a synset is a set of terms to describe the same concept. The terms are searched in a general ontology, thesaurus or lexical database such as WordNet, based on partial matching on the new term in query. In summary, from the vector of original terms that make up the user query $q = \{T_1, T_2,\ldots,T_n\}$ each of the terms of the search vector is taken and concepts are formed so that each concept C is equal to (T, S, H, h). Each concept is equal to the term typed by the user and the semantically related terms that were retrieved from WordNet.
2. **Terms listed in the profile**: In the previous step we obtained a temporary extended search, but not all of these terms should be presented to the user in an auto-complete list. This is why it is necessary to define the order of presentation of the terms, so that to a greater degree they relate to the needs of the user. The aim of this step is just that, to set the order of presentation in relation to the user profile. To achieve this, the user's ***term co-occurrence matrix (matrix S)*** is consulted—the degree of correlation between each term and its related terms (S, H and h) for the current user (U), placing them in order of declining correlation (the most correlated to the least correlated).
   The first item in the drop-down list that is shown to the user is obtained by concatenating the original query without any processing and the term (S, H or h) with the highest degree of correlation. The second is obtained in a similar manner, the original query and the term with the second highest degree of correlation and so on up to a maximum number of terms to be presented on the interface (the model parameter is known as AutoComplete List Size or ACLS). In the event that the user has no information in the S matrix, the drop-down list gives priority to the terms written most recently, adding line by line first the synonyms, then the hyponyms and finally the hypernyms.
3. **Using an external service**: If in Step 1 (Pre-processing and semantic relation) of the model there is no information listed in WordNet, it goes to an external auto complete service, such as that of Google (based on the analysis of query logs of its users, with a focus on collaborative filtering). At this point, and as a future work, the model could incorporate an automatic approach of relevance feedback based on the Top-N documents retrieved (using an automatic relevance feedback based on partial documents).

The ***user profile*** is a fine-grained structure that relates for each user the number of documents reviewed by the user as relevant and irrelevant (N) (user feedback), the number of documents containing a term i (n_i), the number of relevant documents (R) and the number of relevant documents containing the term i (r_i). Moreover, for each document, the presence (1) or absence (0) of terms is recorded. From this information a matrix of term co-occurrence for each user is generated. This co-occurrence matrix, called S, is calculated as shown in Fig 6.

```
01  For each document d Є D do
02      For each term t_i Є d do
03          For each window w_z centered in term t_i do
04              For each term t_j Є w_z where t_j != t_i do
05                  S_i,j = C_i,j * IDF_i * IDF_j
06                  S_j,i = S_i,j
07              End-for
08          End-for
09      End-for
10  End-for
```

Fig 6. Algorithm for generating the term co-occurrence matrix (S).

The correlation factor $C_{i,j}$ is a normalized factor traditionally used in information retrieval [2].

It is defined by (2), where F_i is the frequency of term i, F_j is the frequency of term j and $F_{i,j}$ is the frequency of co-occurrence of terms i and j:

$$\mathrm{idf}_i = \frac{F_{i,j}}{F_i + F_j - F_{i,j}}. \quad (2)$$

The relative importance of a term in information retrieval is given by its IDF (inverse document frequency) value.

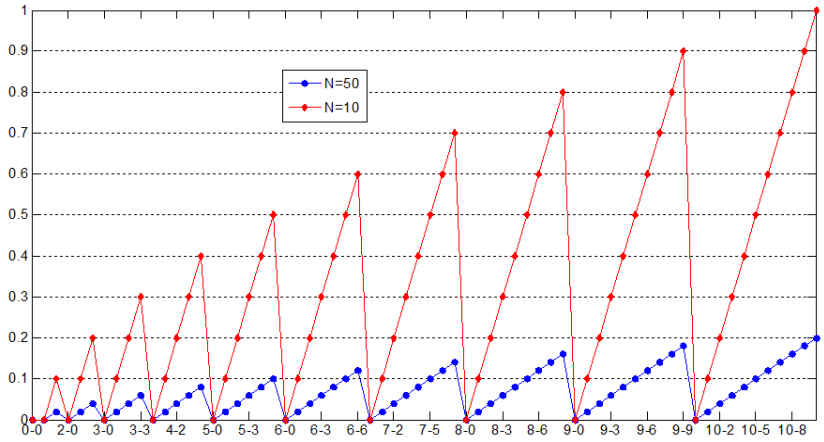

Fig 7. Graph of the IDF function used to calculate the S matrix. The function with N = 10 is shown by the marker in the form of squares and the function with N = 50 is shown by the ovals. The X axis shows different values of n_i and r_i, beginning with (0-0), passing for example through (6,3) and finishing at (10-6). The graph shows values of n_i between 0 and 10 and values of r_i between 0 and 6. For both functions the maximum is achieved when $n_i = r_i$, in this case (6,6) and the minimum when $r_i = 0$, regardless of the value of n_i.

To define this value, a range of formula can be used, e.g. the Robertson and Spärck-Jones (RSJ) proposal [31] one of the most cited in the literature. For our research, the RSJ formula was not suitable for construction of the S matrix. A new function based on formula (3) was defined instead. This IDF function (see Fig 7) defines the importance of a term in relation to the number of documents reviewed by the user (N), the number of documents relevant to the user (R), the number of documents in which the term i appears (n_i) and the number of relevant documents in which the term appears i (r_i):

$$\mathrm{idf_i} = \begin{Bmatrix} \frac{r_i}{N} \text{ si } n_i \leq R \\ \frac{r_i R}{n_i N} \text{ si } n_i > R \end{Bmatrix} \quad (3)$$

The IDF function proposed has a continuous range of values between zero and one [0,1]: zero when the term is not relevant at all and one when it is considered entirely relevant. The degree of relevance is in relation to the range of relevant documents, i.e. if there are many documents reviewed (as in the graph of N=50) and among these the term appears in only a few documents (e.g. 6) and all are relevant, the function has a value of 0.1, compared with a smaller number of documents (for example in the graph N=10), which a value of 0.6 would be obtained.

This IDF function was compared with the traditional Rocchio algorithm in three scenarios (without memory, with memory of the session and a long term memory) using the Communications of the ACM data set, and it obtains better results (see [32] for details).

The term co-occurrence matrix (S) of the user allows the ordered generation of the list of terms that complement those used by the user in the search expansion process as explained above in Step 2 "Terms listed in the profile."

B. Search Results Acquisition and Preprocessing

After performing the search expansion process, there follows the process of ***search results acquisition***. In this step, the query consists of ***key words*** typed by the user (those directly typed by the user and those selected from the auto complete list).

The Acquisition process conducts in parallel (different threads of execution) the collection of results in the various traditional search engines. In the initial model, Google, Yahoo! and Bing are used. As results are returned by the traditional search engines, ***pre-processing of entries*** is carried out. This process includes removing special characters and converting the text to lower case, among others; removing stop words; stemming; and filtering duplicate documents (documents reported concurrently by more than one traditional search engine). In addition, for each document the observed frequency of its terms is calculated and the document is marked as processed.

When all results have been acquired, documents are organized in a ***Terms by Documents Matrix*** using formula (4), which takes into account the relative importance (IDF value) of each term in the retrieved results from traditional search engines. This matrix is the original source of data for the clustering algorithm:

$$w_{i,j} = \left(\frac{F_{i,j}}{\max(F_i)}\right) Log\left(\frac{N}{n_j + 1}\right). \quad (4)$$

C. Cluster construction and labeling

Once the acquisition of search results has finished, the process of ***Cluster construction and Labeling*** follows. This process can be carried out using a variety of existing algorithms, among them Lingo [11], STC [8], SHOC [12],

Dynamic SVD [14]. But, because it should be improve the usefulness of the groupings and clarity of the labeling—as it was mentioned above—, a new algorithm called Agents-WDC was developed.

Agents-WDC is a description-centric algorithm [1] for web document clustering [1] based on Memetic Algorithms (MA) (MAs "are population-based meta-heuristic search methods inspired by both Darwinian principles of natural evolution and Dawkins notion of a meme as a unit of cultural evolution capable of individual learning" [33].) The memetic approach is used to combine a global/local strategy of search in the whole solution space. The *k*-means algorithm was used as a local strategy for improving agents in the MA. Arrival of foreign agents (random generation in evolution process) was used to promote diversity in the population and prevent the population from converging too quickly. The Bayesian Information Criterion (BIC) expressed by formula (5) was used as a fitness function [5, 34]. The evolution process is based on one agent at a time (not of populations) in a specific number of islands and the VSM is used for representing documents in the clustering stage, but in the labeling stage the frequent phrases model is used. Agents-WDC steps can be summarized in Fig 8:

$$BIC = n\,Ln\left(\frac{SSE}{n}\right) + k\,Ln(n),$$
$$SSE = \sum_{j=1}^{k}\sum_{i=1}^{n}\left(P_{i,j}\|x_i - c_j\|\right). \quad (5)$$

Here n is the total number of documents, k is the number of clusters and SSE is the sum of squared error from the similarities of the different clusters; $P_{i,j}$ is 1 if the document x_i belong to cluster j and 0 in other case, and c_j is the centroid of the cluster j.

Initialize algorithm parameters: In this research, the optimization problem lies in minimizing the BIC criteria (Fitness function). Agents-WDC needs the following parameters to be specified: Number of Islands (NI), Population Size (PS), Mutation Rate (MR), Minimum Bandwidth (MinB) and Maximum Bandwidth (MaxB) for mutation operation, Maximum Execution Time (MET) in milliseconds or Maximum Number of Iterations (MNI) to stop the algorithm execution.

Representation and Initialization: In Agents-WDC, each agent has a different number of clusters, a list of centroids, and the objective function value, based on BIC, which depends on the centroids' location in each agent and the number of centroids.

The cluster centers in the agent consist of D x K real numbers, where K is the number of clusters and D is the total number of terms (words in the vocabulary). For example, in three-dimensional data, the agent < [0.3|0.2|0.7], [0.4|0.5|0.1], [0.4|0.1|0.9], [0.0|0.8|0.7], 0.789> encodes centers of four (K value) clusters with a fitness value of 0.789.

1 **Initialize algorithm parameters**.
2 **Repeat** (inner sentences are executed in parallel—each population correspond to an island)
3 **Randomly initialize population** (*PS* agents), which encode cluster centers with different numbers of clusters.
4 **Run** the *k*-means routine for each agent in population.
5 **Calculate fitness** value for each agent in the initial population based on (5).
6 **Repeat**
7 Select pairing parents based on ***roulette wheel.***
8 **Generate one intermediate offspring** by applying genetic operators (***crossover and mutation***) of the paired parents.
9 **Run** the *k*-means routine for the offspring.
10 **Calculate fitness** value for the offspring based on (5).
11 **If** the offspring is invalid (i.e., number of clusters equal to one due to the death units problem in the clustering process) **then** it is replaced with a new agent randomly initialized (***arrival of foreign agents***)
12 **If** the fitness of the offspring is better than the worst agent on population, **then *replace the worst agent*** by the offspring.
13 **Until** MET or MNI is reached.
14 **Select best solution** (agent) in the population.
15 **Until** NI Island finished the process.
16 **Select best solution** (agent) from all islands.
17 **Assign labels to clusters**.
18 **Overlap clusters**.

***k*-means routine** (input: Initial set of centroids)
1 **Repeat**
2 **Re-compute membership** of each document according to current centroids and cosine similarity based on (6).
3 **Update centroids** based on new membership information
4 **Until** no changes in clusters
5 **Return** final set of centroids

Fig 8. Agents-WDC algorithm for web document clustering

Initially, each centroid corresponds to a different document randomly selected in the TDM matrix (Forgy strategy in the *k*-means algorithm [35]).

The initial number of clusters, K value, is randomly calculated from 2 to K_{max} (inclusive), where K is a natural number and K_{max} is the upper limit of the number of clusters and is taken to be $\sqrt{N/2}+1$ (where N is the number of documents in the TDM matrix), which is a rule of thumb used in the clustering literature by many researchers.

Roulette wheel: In step 7, one parent p1 is chosen from the population based on roulette wheel selection [36]. Also, its mate p_2 is chosen by the same process (preventing p_1 equal to p_2).

Crossover and mutation: At step 8 a traditional n-point crossover is used [37]. During crossover, the cluster centers are considered to be indivisible, so crossover points can only lie in between two cluster centers. In this process, just one offspring is generated. After crossover, a low probability of mutation (MR) is applied to the offspring. Uniform mutation between Minimum Bandwidth (MinB) and Maximum Bandwidth (MaxB) (as found in the Harmony Search Algorithm [23]) is applied to the chosen cluster dimension / attribute / term [x = x ± Random (MinB, MaxB)]. When mutation operation generates a value that reaches data boundaries, the mutation value is applied in the opposite way (mirror):

$$Sim(d_i, d_j) = \frac{\sum_{i=1}^{D}\left(W_{i,d_i}W_{i,d_j}\right)}{\sqrt{\sum_{i=1}^{D}\left(W_{i,d_i}\right)^2}\sqrt{\sum_{i=1}^{D}\left(W_{i,d_j}\right)^2}}. \quad (6)$$

Assign labels to clusters: This step corresponds to Step 2 "Frequent Phrase Extraction" in Lingo [11], but in Agents-WDC this method is used for each generated cluster in previous steps. By the above method, some changes were made to the original algorithm. This process works as shown in Fig 9.

Overlap clusters: Finally, each cluster includes documents that fall into other clusters too, if these documents are at a distance of less than or equal to the average distance of the cluster.

01 **Conversion of the representation**: Each document in the current cluster is converted from character-based to word-based representation.

02 **Document concatenation**: All documents in the current cluster are concatenated and a new document with the inverted version of the concatenated documents is created.

03 **Complete phrase discovery**: Right-complete phrases and left-complete phrases are discovered in the current cluster, then the right-complete phrases and left-complete phrases are alphabetically sorted, and then the left- and right-complete phrases are combined into a set of complete phrases.

04 **Final selection**: Terms and phrases whose frequency exceeds the Term Frequency Threshold are selected for the current cluster.

05 **Building of the "Others" label and cluster**: if some documents don't reach the Term Frequency Threshold, then they are sent to the other clusters.

06 **Cluster label induction**: In the current cluster, a Term-document matrix is built. Then, using cosine similarity, the best candidate terms or phrases for the cluster (which optimize SSE) are selected.

Fig 9. Frequent Phrase Algorithm for Labeling

IV. Experiments

Measuring the clustering performance of a document clustering algorithm is a complex issue. There are many different approaches and no standard methodology.

In general, there are two main categories of evaluation: internal quality (based on objective functions without reference to the output, this is the least used) and external quality (which evaluates the output clustering). External quality assessment can be further divided into gold-standard, task-oriented and user evaluation.

In gold-standard evaluation, results of the algorithm are compared with a pre-determined ideal clustering. In task-oriented evaluation, a performance analysis of a particular part of an algorithm is done.

External evaluation using gold-standard evaluation and user evaluation is the most common approach for evaluating the performance of web document clustering algorithms [38]. Thus, in this research this is the approach that has been used.

A. Datasets for Assessment

The Open Directory Project (or DMOZ) is commonly used as a neutral third party classifier, using human editors to classify manually and store thousands of websites. In this research a total of fifty datasets were randomly built. Datasets are available online at www.unicauca.edu.co/~ccobos/wdc/wdc.htm.

On average, datasets have 129.2 documents, 6 topics and 643.9 terms. Fig 10 shows different views of the datasets content and Table 1 shows detailed information from each dataset.

B. Parameters and Measures

Parameter values in Agents-WDC were equal for all datasets. NI equal to 2, PS equal to 5, MR equal to 0.5%, MinB equal to 0,0005, MaxB equal to 0.005, and MNI between 0 and 40 (depending on the experiment). Kmax value was equal to $\sqrt{N/2}+1$, where N is the number of documents.

There are many different methods proposed for measuring the quality of a generated clustering compared to an ideal clustering. Three of the best known are precision, recall and F-measure, commonly used in information retrieval and classification tasks [9].

In this research, the weighted Precision, weighted Recall and weighted F-measure (the harmonic means of precision and recall) measures are used to evaluate the quality of solution.

Given a collection of clusters, $\{C_1, C_2, \ldots C_k\}$, to evaluate its weighted Precision, weighted Recall and weighted F-measure with respect to a collection of ideal clusters $\{C_1^i, C_2^i, \ldots C_h^i\}$, these steps are followed: (a) find for each ideal cluster C_n^i a distinct cluster C_m that best approximates it in the collection being evaluated, and evaluate $P(C, C^i)$, $R(C, C^i)$, and $F(C, C^i)$ as defined by (7) and (8). (b) Calculate the weighted Precision (P), weighted Recall (R) and weighted F-measure (F) based on (9):

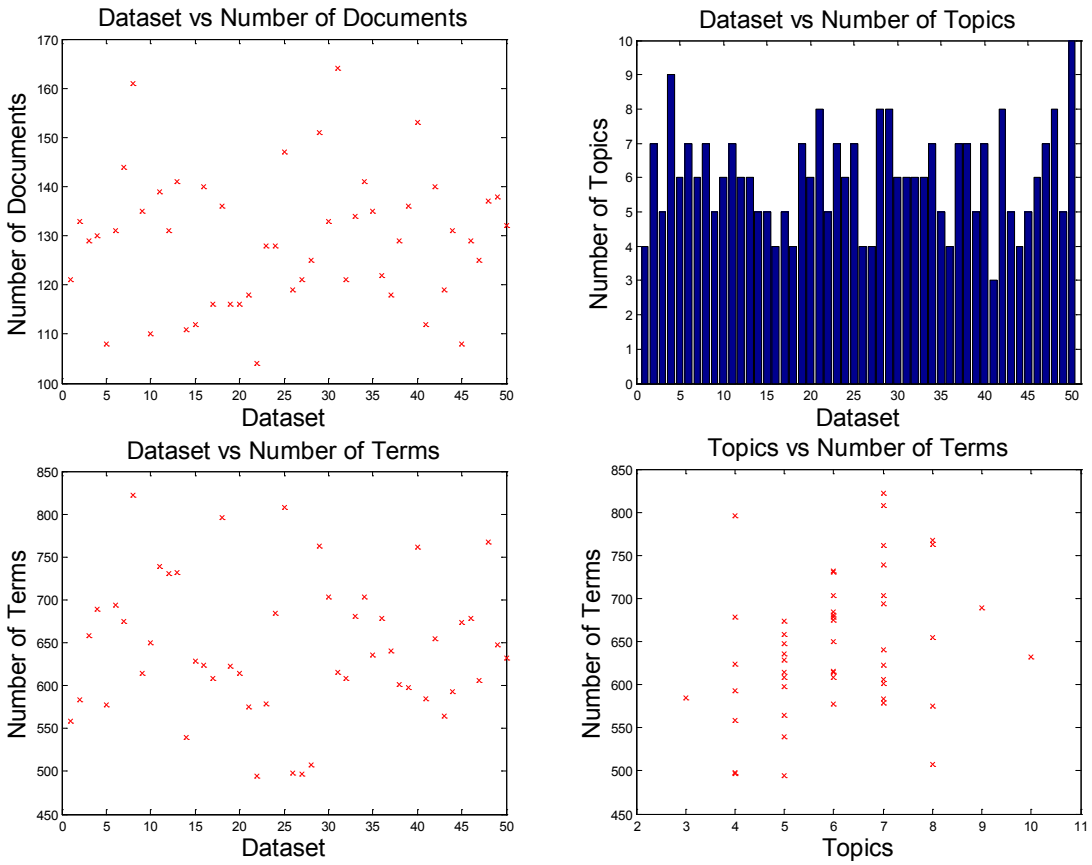

Fig 10. Datasets used for evaluation vs. documents, topics and number of terms

TABLE 1
GENERAL DESCRIPTION OF DATASETS USED FOR EVALUATION
(P STANDS FOR PRECISION AND R FOR RECALL USING SUPPORT VECTOR MACHINE)

Dataset	Documents	Topics	Attributes	P	R	Dataset	Documents	Topics	Attributes	P	R
1	121	4	559	96.865	96.694	26	119	4	498	89.428	88.235
2	133	7	583	90.183	89.474	27	121	4	497	90.062	88.43
3	129	5	658	94.358	93.023	28	125	8	507	90.757	89.6
4	130	9	689	87.601	83.846	29	151	8	763	90.694	89.404
5	108	6	578	84.453	81.481	30	133	6	703	87.352	85.714
6	131	7	694	94.252	93.893	31	164	6	616	96.048	95.732
7	144	6	675	93.984	93.056	32	121	6	609	91.982	90.909
8	161	7	822	92.501	91.304	33	134	6	681	87.574	88.06
9	135	5	614	92.131	91.111	34	141	7	703	91.357	89.362
10	110	6	650	92.629	89.091	35	135	5	636	97.9	97.778
11	139	7	739	94.427	93.525	36	122	4	679	96.006	95.902
12	131	6	731	90.037	89.313	37	118	7	641	85.089	79.661
13	141	6	732	67.174	66.429	38	129	7	601	88.219	86.822
14	111	5	540	95.118	93.694	39	136	5	598	95.18	94.853
15	112	5	629	94.943	94.643	40	153	7	761	89.747	89.542
16	140	4	624	93.786	93.571	41	112	3	585	92.695	91.071
17	116	5	609	92.92	92.241	42	140	8	655	87.875	87.143
18	136	4	796	94.443	94.118	43	119	5	564	94.624	94.118
19	116	7	623	94.31	93.966	44	131	4	593	94.445	93.13
20	116	6	614	88.61	84.483	45	108	5	674	80.426	80.556
21	118	8	575	83.678	78.814	46	129	6	679	91.334	89.147
22	104	5	495	93.477	93.269	47	125	7	606	87.263	86.4
23	128	7	579	92.07	90.625	48	137	8	767	91.094	90.511
24	128	6	684	86.416	85.156	49	138	5	648	89.577	88.406
25	147	7	808	88.835	87.075	50	132	10	632	88.509	76.515

TABLE 2
PRECISION, RECALL AND F-MEASURE IN K-MEANS, BISECTING K-MEANS, STC, LINGO AND AGENTS-WDC

Algorithm	Time	Precision	Recall	F-Measure	Number of Fitness evaluations	Ideal number of topics (average)	Obtained number of clusters (average)
k-means	1.0 ± 0.262	75.08 ± 13.39	55.78	63.59 ± 9.06	–	6.02 ± 1.464	**8.72** ± 1.386
Bisecting k-means	0.53 ± 0.009	70.49 ± 10.22	40.08	49.21 ± 4.75	–	6.02 ± 1.464	11.94 ± 1.23
STC	**0.02** ± 0.003	77.68 ± 9.59	45.18	56.71 ± 8.857	–	6.02 ± 1.464	15.97 ± 0.24
Lingo	0.68 ± 0.177	**80.34** ± 4.419	29.53	43.00 ± 4.761	–	6.02 ± 1.464	34.46 ± 1.784
Agents-WDC	1.55 ± 0.38	79.72 ± 13.95	**59.29**	**67.43** ± 9.979	12 = 7 + PS	6.02 ± 1.464	**8.96** ± 2.185
Agents-WDC	1.78 ± 0.45	**81.46** ± 11.15	**61.17**	**69.29** ± 9.464	14 = 9 + PS	6.02 ± 1.464	**8.80** ± 2.000
Agents-WDC	1.85 ± 0.56	**82.63** ± 9.311	**61.07**	**69.72** ± 8.886	16 = 11 + PS	6.02 ± 1.464	**8.90** ± 1.632
Agents-WDC	5.57 2.073	**88.85** ± 8.731	63.95	73.53± 7.698	45 = 40 + PS	6.02 ± 1.464	9.44 ± 2.21

$$P(C, C^i) = \frac{|C \cap C^i|}{|C|}, \qquad R(C, C^i) = \frac{|C \cap C^i|}{|C^i|}; \tag{7}$$

$$F(C, C^i) = \frac{2P(C, C^i)R(C, C^i)}{P(C, C^i) + R(C, C^i)}; \tag{8}$$

$$P = \frac{1}{T}\sum_{j=1}^{h} |C_j^i| P(C_m, C_j^i); \quad R = \frac{1}{T}\sum_{j=1}^{h} |C_j^i| R(C_m, C_j^i); \quad F = \frac{2PR}{P+R}; \quad T = \sum_{j=1}^{h} |C_j^i|. \tag{9}$$

Here, C is a cluster of documents and cluster C^i is an ideal cluster of documents.

C. Results with Datasets

The algorithm was compared with a version of *k*-means (it executes several solutions of *k*-means and selects the best solutions based on BIC criteria), Bisecting K-means, STC and Lingo (last three algorithms are provided by the free open source Carrot2 Document Clustering at www.carrot2.org and were used with its default values).

Using an on-line scenario (with 2.0 seconds as a maximum time of execution and without query expansion support), algorithms was executed 30 times over each dataset and the averages were calculated to show them as results. These promising results are shown in Table 2. High values of Precision, Recall, and F-Measure are desirable.

In Table 2, the best results are presented when Agents-WDC is executed 11 iterations (approximately 1.85 seconds in a desktop computer with windows vista of 32 bits, 4 GB of RAM and Intel Pentium Dual CPU at 2.16 GHz. Time has a linear correlation with the iterations, equivalent in this setting to $Time = 0{,}1116 * iterations + 0{,}1395$). Also, Agents-WDC reports very competitive results from 7 iterations (1.55 seconds of execution time). Recall and F-Measure is always better with Agents-WDC algorithm.

Lingo and STC reports very good precisions with a low time of execution, but Recall and F-Measure are too far from Agents-WDC results. The recall distance between Agents-WDC and STC is around 15% on Recall and around of 30% against Lingo.

Lingo reported the lowest rate of dispersion in precision, while Agents-WDC reported in 1.85 seconds more than twice that value. Although in Agents-WDC the precision variation decreases over the iterations, this is an issue that should be studied further.

Another important difference between Agents-WDC, Bisecting *k*-means, STC and Lingo is the number of clusters. Agents-WDC always defines a better value of K (number of clusters). In Lingo and STC with an average of 28 and 9 extra clusters respectively, results of precision can be biased. Therefore, the research group plans to use another kind of metrics to compare results of STC and Lingo, for example BCubed Precision and BCubed Recall [39].

In Fig 11, curves of precision, recall and f-measure through different number of generations are shown. All values increasing with the number of generations. Therefore, when users can wait for results, Agents-WDC organized in better way clusters of documents and proved the best option. BIC with cosine similarity is a good option for web document clustering because precision and recall both increase when Agents-WDC optimizes BIC ($Precision = 6.2759 * \ln(Generations) + 65.015$ with $R^2 = 95.43\%$), but in some generations (e.g. 4 to 6 generations) this positive relation fails. Thus, the research group plans to define a better fitness function for evolutionary algorithms in web document clustering based on genetic programming.

Further analysis showed that in general Agents-WDC increases the quality of cluster (based on precision and recall) when it uses more generations regardless of the number of documents, number of topics, or number of attributes in the dataset. Some datasets, though, do not comply with this rule.

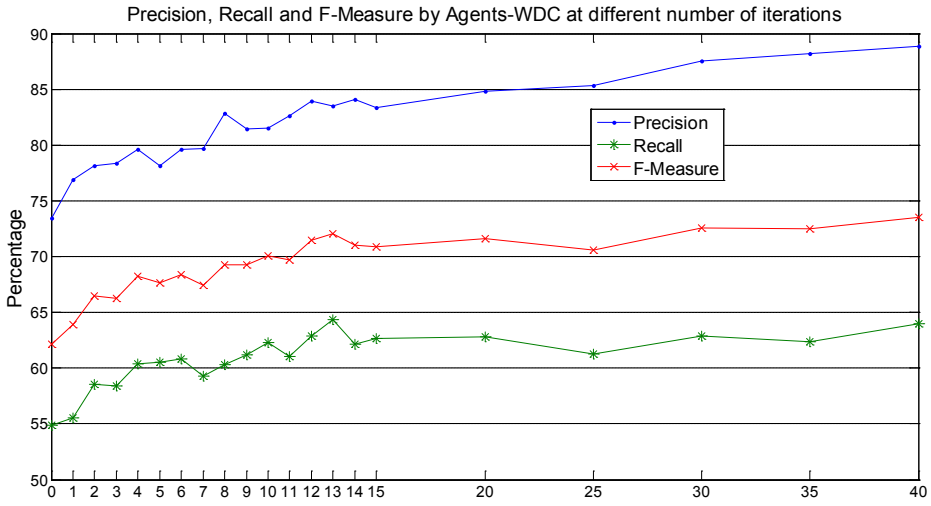

Fig 11. Precision, Recall and F-Measure for Agents-WDC through different periods of time

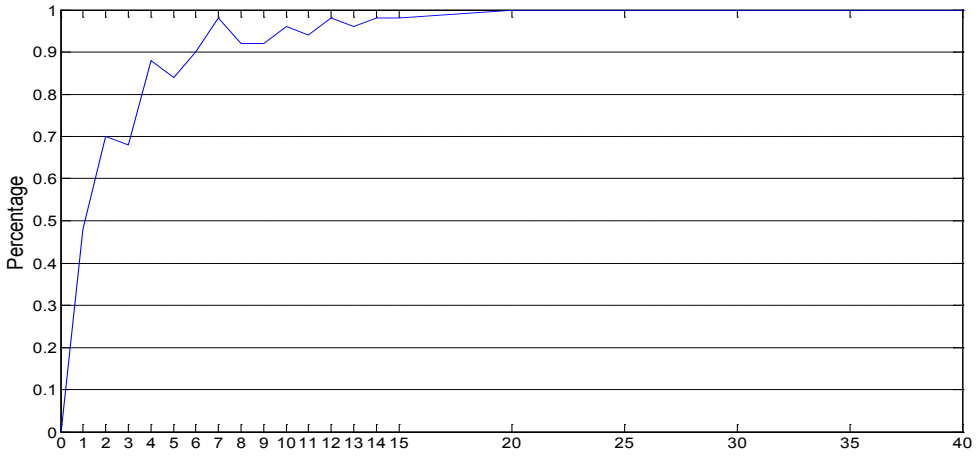

Fig 12. Effectiveness of new solutions generated at different number of iterations

This situation reinforces the need for defining a new objective function as was mentioned above, but also implies the need to analyze the impact of noise on the *k*-means algorithm, and the need to use other local optimization algorithms.

New solutions (agents) generated (using the selection, crossover, mutation and replace operators from Agents-WDC) increase its effectively over iterations. Fig 12 shows a 48% of effectively of the new solution in the first iteration, i.e. new solution is better than other solutions in population. Next, the effectiveness increases to 70% in second iteration, then it increases to 90% in sixth iteration, and finally is around 100% over the twelfth iteration.

Agents-WDC also provided better cluster labels than Bisecting k-means, STC and Lingo. For example, Table 3 shows labels generated by all algorithms for dataset1 with 4 non-overlapping topics. Note that the clusters generated and the order in which they are generated are different between the algorithms.

It is clear that Agents-WDC and STC generate best labels, while Lingo generates longer phrases and Bisecting *k*-means generates a long list of terms for each cluster. Lingo's long labels, while expressive, can be too specific and not always meaningful (e.g. "Shows how to Use"). Lingo only classified 74 out of 121 documents, much fewer than Agents-WDC, STC and Bisecting *k*-means. Agents-WDC favors labels with specific meanings closely related to documents in the cluster.

D. User Evaluation

Based on [40], a user-based evaluation method was used to assess the clustering results produced by the model (on-line scenario with 40 users) when data sources are Google, Yahoo! and Bing. For a set of groups created in response to a single query, the user answered whether or not:

TABLE 3
LABELS GENERATED BY BISECTING K-MEANS, STC, LINGO AND AGENTS-WDC OVER DATASET 1

Real category in DMOZ	Bisecting k-means	STC	Lingo	Agents-WDC (1.2seg)
Top/Business/Textiles_and_Nonwovens/Fibers/Wholesale_and_Distribution (16), Top/Health/Conditions_and_Diseases/Cancer/Breast (22), Top/Computers/Programming/Languages/Regular_Expressions (34), Top/Shopping/Food/Sweeteners/Honey (49)	Cancer, Breast, Male(13), Nonwovens, Polyester, English(13), Regular, Allows, Perform(12), Regular, Show, Windows(12), Beeswax, Candles, Overview(10), Diagnosis, Symptoms, Prevention(10), Raw, Business, Unprocessed(10), New, Royal, Zealand(9), Examines, Mitchell, Scott(6), Regex, JavaScript, Tester(6), Beeswax, Sioux, Clinically-oriented(5), Forest, Ordering, Trees(5), National, Centre, Ovarian(4), Wax, Cooperative, Indiana(4), Links, Representatives, Sites(2)	Regular Expressions(31), Treatment, Diagnosis, Breast Cancer(17), Yarns, Natural and Man-made Fibers(10), Honey(44), Nonwovens, Staple Fiber, Polyester(12), Regular Expression(8), Polyester Staple Fiber, Nonwovens Production, Yarn Spinning(5), Information(19), Natural(17), Beeswax(15), Fibers(13), Products(13), Raw(11), Raw Honey(6), Offers(10), Other Topics(6)	Breast Cancer(9), Beeswax Candles(6), Overview(5), Produces Raw(5), Business(4), English and German(4), Sold(4), Windows(4), Introduction to Regular(3), New Zealand(3), North Dakota(3), Organic(3), Parts(3), Polyester and Polyamide Filaments(3), Power of Regular(3), Shows how to Use(3), Sizes(3), Commercial(2), Farmer Market Schedule(2), Flower(2), Gift Baskets(2), Help(2), International Merchants(2), Jar(2), Male Breast Cancer(2), National Breast and Ovarian Cancer Centre(2), Regex(2), Risk Factors(2), Scott Mitchell Examines(2), Search(2), Short Video and Details of Symptoms(2), Source Code(2), Visual(2), Other Topics(47)	Information Overview (14), Produces Raw Honey (31), Regular Expressions This Article (31), Bee Pollen (12), Details Of Symptoms Causes Diagnosis Treatment Prevention (16), Natural and Man Made Fibers Yarns (17)

– (Q1: concise and meaningful cluster labels) the cluster label for each group is in general representative of the cluster content (much: R3, little: R2, or nothing: R1). Concise and meaningful cluster labels help users to decide which groups should review.
– (Q2: Usefulness of clusters) the cluster description and content is useful (R3), moderately useful (R2) or useless (R1). Usefulness of clusters is a general assessment (quality of labels and content) only for those groups that users decided relevant to query.

Then, for each document in each cluster, the user answered whether or not:

– (Q3) the document (snippet) matches with the cluster (very well matching: R3, moderately matching: R2, or not-matching: R1), A very well matching document would contain exactly the information suggested by the cluster label. A moderately matching document would still be somehow related to the group's topic. A non-matching document, even though it might contain several words from the group's label, would be completely irrelevant to its cluster.
– (Q4) the document relevance (order or rank) in the cluster was adequate (adequate: R3, moderately suitable: R2, or inadequate: R1). The most relevant documents should appear in the top of the list of group outcomes. This makes the user spend less time to solve their information needs.

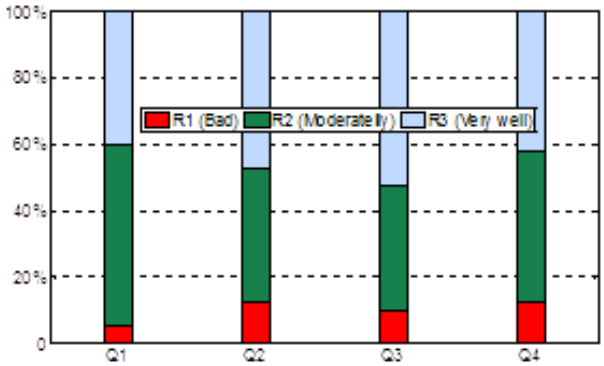

Fig 13. General results for the four questions (TDM-BIC-FPH)

General results of Agents-WDC are shown in Fig 13. Most user responses (90%) are R3 or R2. Therefore, results are very promising and it is necessary to do a set of very controlled experiments with more users, in order to generalize results. In summary, most of the users find that: cluster labels are representative, clusters are useful and documents are well organized in each cluster.

V. CONCLUSIONS AND FUTURE WORK

The proposed personalized web document clustering model allows users to define better queries, based on WordNet (semantic similarity of terms) and a user profile (order based on the new IDF function and correlation of terms). In the description of the model, the query expansion process,

acquisition of search results from traditional web search engines, the preprocessing of input data, a web clustering algorithm based on a memetic approach, and a proposal for cluster labeling are all detailed. All of these processes were easily modeled with agents.

The Clustering and Labeling Agent uses the Agents-WDC algorithm. This algorithm is a web document clustering algorithm based on Memetic Algorithms (global/local search strategy) and the *k*-means algorithm (local solution improvement strategy) with the capacity of automatically defining the number of clusters. Agents-WDC shows promising experimental results in standard datasets. Comparison with *k*-means, Bisecting *k*-means, STC and Lingo show Agents-WDC is a better algorithm for web document clustering in both on-line and off-line scenarios.

The Bayesian Information Criterion (BIC) with cosine similarity is a good option for web document clustering because precision and recall both increase when Agents-WDC algorithm evolve, but in some cases this positive relation fail.

New solutions (agents) generated in Agents-WDC algorithm based on roulette selection, a traditional n-point crossover, uniform mutation, and local improvement with k-means show a high rate of success (around 90% from fifth iteration) in the evolutionary process.

Agents-WDC uses two document representations models, initially it uses vector space model in the clustering process and then it uses full text for the labels creation process. This combination improves quality of cluster labels and the general quality of the clustering process.

There follow some suggestions for future work: applying the model to several datasets (other datasets based on DMOZ, Google results, Yahoo! results, among others) in on-line and off-line scenarios; evaluate Agents-WDC using BCubed Precision and BCubed Recall and compare results with Lingo, STC and other web document clustering algorithms. Define a new fitness function for evolutionary algorithms in web document clustering, using for example genetic programming.

Evaluate TopicSearch alongside other traditional and evolutionary algorithms for web document clustering; making use of WordNet to work with concepts (Concept-Document Matrix) instead of terms (Term-Document Matrix) and comparing the results; evaluating the entire model with a lot of users in different contexts; and evaluating the impact of query expansion over the time.

ACKNOWLEDGMENT

The work in this paper was supported by a Research Grant from the University of Cauca under Project VRI-2560 and the National University of Colombia.

REFERENCES

[1] C. Carpineto, *et al.*, "A survey of Web clustering engines," *ACM Comput. Surv.,* vol. 41, pp. 1-38, 2009.

[2] R. Baeza-Yates, A. and B. Ribeiro-Neto, *Modern Information Retrieval*: Addison-Wesley Longman Publishing Co., Inc., 1999.

[3] C. Carpineto, *et al.*, "Evaluating subtopic retrieval methods: Clustering versus diversification of search results," *Information Processing & Management,* vol. 48, pp. 358-373, 2012.

[4] K. Hammouda, "Web Mining: Clustering Web Documents A Preliminary Review," ed, 2001, pp. 1-13.

[5] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*: Prentice-Hall, Inc., 1988.

[6] M. Steinbach, *et al.*, "A comparison of document clustering techniques," in *KDD workshop on text mining*, Boston, MA, USA., 2000, pp. 1-20.

[7] Y. Li, *et al.*, "Text document clustering based on frequent word meaning sequences," *Data & Knowledge Engineering,* vol. 64, pp. 381-404, 2008.

[8] Z. Oren and E. Oren, "Web document clustering: a feasibility demonstration," presented at the Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, Melbourne, Australia, 1998.

[9] M. Mahdavi and H. Abolhassani, "Harmony K-means algorithm for document clustering," *Data Mining and Knowledge Discovery,* vol. 18, pp. 370-391, 2009.

[10] P. Berkhin, *et al.*, "A Survey of Clustering Data Mining Techniques," in *Grouping Multidimensional Data*, ed: Springer-Verlag, 2006, pp. 25-71.

[11] S. Osiński and D. Weiss, "A concept-driven algorithm for clustering search results," *Intelligent Systems, IEEE,* vol. 20, pp. 48-54, 2005.

[12] D. Zhang and Y. Dong, "Semantic, Hierarchical, Online Clustering of Web Search Results," in *Advanced Web Technologies and Applications*, ed, 2004, pp. 69-78.

[13] B. Fung, *et al.*, "Hierarchical document clustering using frequent itemsets," in *Proceedings of the SIAM International Conference on Data Mining*, 2003, pp. 59-70.

[14] G. Mecca, *et al.*, "A new algorithm for clustering search results," *Data & Knowledge Engineering,* vol. 62, pp. 504-522, 2007.

[15] F. Beil, *et al.*, "Frequent term-based text clustering," in *KDD '02: International conference on Knowledge discovery and data mining (ACM SIGKDD)*, Edmonton, Alberta, Canada, 2002, pp. 436-442.

[16] L. Jing, "Survey of Text Clustering," ed, 2008.

[17] W. Song, *et al.*, "Genetic algorithm for text clustering using ontology and evaluating the validity of various semantic similarity measures," *Expert Systems with Applications,* vol. 36, pp. 9095-9104, 2009.

[18] L. Xiang-Wei, *et al.*, "The research of text clustering algorithms based on frequent term sets," in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, 2005, pp. 2352-2356 Vol. 4.

[19] A. K. Jain, *et al.*, "Data clustering: a review," *ACM Comput. Surv.,* vol. 31, pp. 264-323, 1999.

[20] S. Osiński and D. Weiss, "Carrot 2: Design of a Flexible and Efficient Web Information Retrieval Framework," in *Advances in Web Intelligence*, ed, 2005, pp. 439-444.

[21] X. Wei, *et al.*, "Document clustering based on non-negative matrix factorization," presented at the Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, Toronto, Canada, 2003.

[22] Z. Zhong-Yuan and J. Zhang, "Survey on the Variations and Applications of Nonnegative Matrix Factorization," in *ISORA'10: The Ninth International Symposium on Operations Research and Its Applications*, Chengdu-Jiuzhaigou, China, 2010, pp. 317–323.

[23] Z. Geem, *et al.*, "A New Heuristic Optimization Algorithm: Harmony Search," *Simulation,* vol. 76, pp. 60-68, 2001.

[24] R. Forsati, *et al.*, "Hybridization of K-Means and Harmony Search Methods for Web Page Clustering," in *WI-IAT '08: IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2008, pp. 329-335.

[25] M. Mahdavi, *et al.*, "Novel meta-heuristic algorithms for clustering web documents," *Applied Mathematics and Computation,* vol. 201, pp. 441-451, 2008.

[26] W. Song and S. Park, "Genetic Algorithm-Based Text Clustering Technique," in *Advances in Natural Computation*, ed, 2006, pp. 779-782.

[27] C. Cobos, *et al.*, "Web document clustering based on Global-Best Harmony Search, K-means, Frequent Term Sets and Bayesian

Information Criterion," in *2010 IEEE Congress on Evolutionary Computation (CEC)*, Barcelona, Spain, 2010, pp. 4637-4644.

[28] C. Cobos*, et al.*, "Web Document Clustering based on a New Niching Memetic Algorithm, Term-Document Matrix and Bayesian Information Criterion," in *2010 IEEE Congress on Evolutionary Computation (CEC)*, Barcelona, Spain, 2010, pp. 4629-4636.

[29] C. Manning*, et al.* (2008). *Introduction to Information Retrieval*. Available: http://www-csli.stanford.edu/~hinrich/information-retrieval-book.html

[30] L. Yongli*, et al.*, "A Query Expansion Algorithm Based on Phrases Semantic Similarity," presented at the Proceedings of the 2008 International Symposiums on Information Processing, 2008.

[31] S. E. Robertson and K. Sparck-Jones, "Relevance weighting of search terms," in *Document retrieval systems*, ed: Taylor Graham Publishing, 1988, pp. 143-160.

[32] C. Cobos*, et al.*, "Algoritmos de Expansión de Consulta basados en una Nueva Función Discreta de Relevancia," *Revista UIS Ingenierías,* vol. 10, pp. 9-22, Junio 2011.

[33] Q. H. Nguyen*, et al.*, "A study on the design issues of Memetic Algorithm," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, 2007, pp. 2390-2397.

[34] A. Webb, *Statistical Pattern Recognition, 2nd Edition*: {John Wiley & Sons}, 2002.

[35] S. J. Redmond and C. Heneghan, "A method for initialising the K-means clustering algorithm using kd-trees," *Pattern Recognition Letters,* vol. 28, pp. 965-973, 2007.

[36] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: The MIT Press, 1999.

[37] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*: Addison-Wesley Longman Publishing Co., Inc., 1989.

[38] T. Matsumoto and E. Hung, "Fuzzy clustering and relevance ranking of web search results with differentiating cluster label generation," in *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, 2010, pp. 1-8.

[39] E. Amigó*, et al.*, "A comparison of extrinsic clustering evaluation metrics based on formal constraints," *Inf. Retr.,* vol. 12, pp. 461-486, 2009.

[40] S. Osiński, "An Algorithm for clustering of web search results," Master, Poznań University of Technology, Poland, 2003.