Available online at www.sciencedirect.com
ScienceDirect
Information Processing and Management 43 (2007) 752–768
INFORMATION PROCESSING & MANAGEMENT
www.elsevier.com/locate/infoproman

Topic discovery based on text mining techniques

Aurora Pons-Porrata [a], Rafael Berlanga-Llavori [b,*], José Ruiz-Shulcloper [c]

[a] Center of Pattern Recognition and Data Mining, Universidad de Oriente, Patricio Lumumba s/n, Santiago de Cuba 90500, Cuba
[b] Computer Science, Universitat Jaume I, Avda. Vicent Sos Banyat, Campus del Riu Sec s/n, E-12071 Castellón, Spain
[c] Advanced Technologies Application Center, 7ma, No. 21812, Siboney, C. Habana, Cuba

Received 10 March 2006; received in revised form 31 May 2006; accepted 3 June 2006
Available online 6 September 2006

Abstract

In this paper, we present a topic discovery system aimed to reveal the implicit knowledge present in news streams. This knowledge is expressed as a hierarchy of topic/subtopics, where each topic contains the set of documents that are related to it and a summary extracted from these documents. Summaries so built are useful to browse and select topics of interest from the generated hierarchies. Our proposal consists of a new incremental hierarchical clustering algorithm, which combines both partitional and agglomerative approaches, taking the main benefits from them. Finally, a new summarization method based on Testor Theory has been proposed to build the topic summaries. Experimental results in the TDT2 collection demonstrate its usefulness and effectiveness not only as a topic detection system, but also as a classification and summarization tool.

Keywords: Hierarchical clustering; Text summarization; Topic detection

1. Introduction

The popularity of Internet has caused an ever-increasing amount of textual documents (Web pages, news, scientific papers, etc.). This information explosion has led to a growing challenge for Information Retrieval systems to efficiently and effectively manage and retrieve this information. The standard formulation of the information access problem presumes a query, the user's expression of an information need. However, many times it is difficult, if not impossible, to formulate such a query. Suppose, for example, a user wants to find out what events happened during a given month and/or at a given place. In this case, the information need is too vague to be described as a single topic and the user does not know the topics of her interest. Indeed, the user may wish to browse over a set of discovered topics extracted from the document collection at hand.

Text Mining, also known as Intelligent Text Analysis, Text Data Mining or Knowledge Discovery in Text (Feldman & Dagan, 1995), refers generally to the process of extracting interesting and non-trivial information

* Corresponding author. Tel.: +34 964728367; fax: +34 964728435.
E-mail addresses: aurora@app.uo.edu.cu (A. Pons-Porrata), berlanga@lsi.uji.es (R. Berlanga-Llavori), jshulcloper@cenatav.co.cu (J. Ruiz-Shulcloper).

0306-4573/$ - see front matter
doi:10.1016/j.ipm.2006.06.001

and knowledge from unstructured text collections. Clustering is a useful technique in Text Mining for discovering interesting data distribution and patterns in the underlying data. Using this technique, interesting structures or clusters can be found directly from the data without relying upon any background knowledge.

Current clustering techniques can be broadly classified into two categories: partitional and hierarchical. Hierarchical clustering algorithms are ideal tools for their interactive visualization and browsing as they provide data-views that are consistent, predictable, and with different granularity levels. Scatter/Gather (Cutting, Pedersen, Karger, & Tukey, 1992) firstly introduced document clustering as a document browsing method, which is also the focus of our work.

There is no doubt that a document collection organized into a hierarchy is very helpful for users. However, it is not enough. Users also need to determine at a glance whether clusters are of their interest by means of some kind of summary. In the literature, the problem of summarizing a set of input documents, called *multidocument summarization*, has received much attention lately (e.g. Mani & Bloedorn, 1997; Barzilay, Elhadad, & McKeown, 2002). Basically, a multidocument summarization system tries to determine which sentences must be included in the summary, and then how to organize them to make the summary comprehensible.

In this context, our research focuses on two issues: (1) how to discover the structure of topics reported in a news collection, that is, how to identify not only the topics but also the subtopics they comprise and (2) how to properly summarize these topics and their subtopics.

For example, Fig. 1 presents part of the structure of the "Current conflict with Iraq" topic from the TDT2 collection as users would desire to obtain. In the first level of the hierarchy, incoming documents are grouped into small subtopics. In the upper levels, composite topics are successively built upon them. Furthermore, a brief description of each topic is included to help users to browse on the hierarchy. An example of the hierarchy produced by our system is shown in Fig. 6.

In this paper, we present a new incremental hierarchical clustering algorithm. It uses a multilayered clustering to produce the hierarchy. Each level of the hierarchy is generated by using a partitional clustering routine over the cluster representatives of the previous level.

Moreover, we propose a novel and effective algorithm that provides a summary for each hierarchy topic. It is based on the *Testor Theory* (Lazo-Cortés, Ruiz-Shulcloper, & Alba-Cabrera, 2001) defined in the field of Pattern Recognition. Contrary to other multidocument summarization methods, the selection of sentences is based on the frequent discriminating terms of each cluster, which can be efficiently computed.

It is worth mentioning that, in our method, we assume that no previous knowledge about the collection exists, and therefore no training examples are available to supervise the classification of documents. As a consequence, all the processes involved in the proposed method (i.e. topic discovery and summary construction) must be completely unsupervised. Additionally, as our main focus is on web-based applications, we also require incremental clustering so that whenever a new document arrives to the collection it is not necessary to start from scratch.

The proposed method can be also useful for the analysis of large document collections, mainly in the following tasks:

Exploration of document contents. Starting from the top level of the hierarchy, the user can browse and select the topics of her interest. Thus, she has a global view of the collection at any time according to its current contents. Such a global view can change along time as new documents are included in the collection.

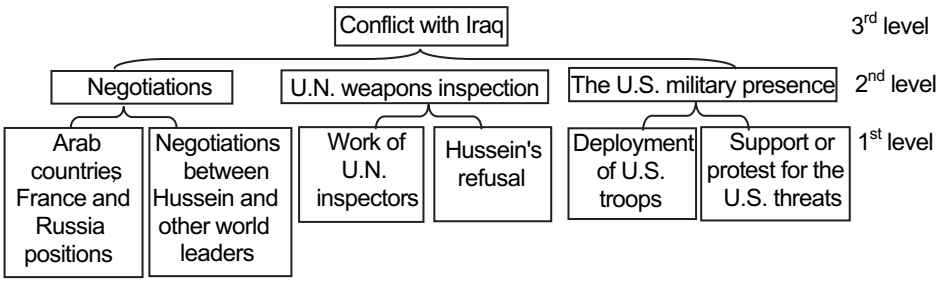

Fig. 1. Structure for the "Current conflict with Iraq" topic.

Topic detection. The *topic detection* problem (Allan, Carbonell, Doddington, Yamron, & Yang, 1998) consists of determining for each incoming document, whether it reports on a new topic, or it belongs to some previously detected topic. A topic detection system forms topic-based clusters of documents. The clusters in our top level of the hierarchy can be seen as detected topics in a stream of news. Also, the lower levels of the hierarchy can present topics with different detail levels.

Multidocument summarization. Each topic of the hierarchy is summarized taking into account the discriminating terms. These summaries can help users to find relevant documents effortless. They can be also used for Information Retrieval purposes (e.g. find the clusters related to a user query).

Dimensionality reduction. The relevant terms extracted with our summarization method can be used to represent the documents in a vector space with less number of dimensions without affecting the clustering quality.

The remainder of the paper is organized as follows: Section 2 introduces some related approaches. Section 3 presents the global architecture of our method, and describes both the representation of documents taking into account their spatio-temporal components, and a document similarity measure for them. Section 4 describes an incremental hierarchical clustering algorithm to discover topics and their subtopics. Section 5 presents the cluster description method to build topic summaries, and Section 6 shows the experiments carried out. Finally, conclusions are presented in Section 7.

2. Related work

This section describes the main works in the literature that concern with hierarchical and incremental clustering algorithms, but that also provide some mechanism to obtain summaries from the generated cluster hierarchies.

Hierarchical clustering algorithms produce a nested sequence of partitions, with a single all-inclusive cluster at the top and singleton clusters of individual documents at the bottom. The result of these algorithms can be viewed as a tree, called a dendrogram. The hierarchical methods can be further divided into agglomerative or divisive algorithms. Hierarchical agglomerative clustering (HAC) methods (e.g. single-link, complete-link, average-link) start with each document in a cluster of its own, iterate by merging the two closest clusters at each step, and finish when some halting criterion is reached. HAC methods differ from the procedure to compute the similarity between clusters and the halting criterion. On the contrary, divisive hierarchical clustering algorithms use a repeated cluster bisectioning approach (Steinbach, Karypis, & Kumar, 2000; Zhao & Karypis, 2004). In these methods, all documents are initially partitioned into two clusters. Then, one non-singleton cluster is selected to be further bisected. This process is repeated until all clusters are singleton.

One of the advantages of partitional clustering algorithms is that they use global information of the collection (e.g. entropy, term distribution, etc.), when partitioning it into clusters. On the other hand, agglomerative clustering algorithms can generate small and reasonably cohesive clusters, a task in which partitional algorithms may fail. However, the agglomerative approaches may take wrong decisions at earlier stages of the clustering process, which tend to be multiplied as the agglomeration of clusters progresses.

In this context, hybrid approaches attempt to combine both the global view used in partitional clustering algorithms and the local view used in agglomerative algorithms. Among them, Scatter/Gather (Cutting et al., 1992) and constrained agglomerative algorithms (Zhao & Karypis, 2005) are two remarkable examples.

Scatter/Gather is a well-known algorithm which has been proposed for a document browsing system based on clustering. It applies a hierarchical agglomerative clustering algorithm on a small sampled dataset to determine an initial clustering which is then refined using the k-means algorithm. The algorithm works in a local manner on small groups of documents rather than trying to deal with the entire corpus globally to reduce the time complexity. This algorithm puts limits on how many clusters can be formed and it is not deterministic, that is, repeated calls to this algorithm on the same corpus may produce different partitions. Scatter/Gather summarizes document clusters by meta-documents containing profiles of topical words and the most typical titles. Topical words are those that occur most frequently in a cluster, and typical titles are those with the highest similarity to the cluster's centroid. Together, the topical words and typical titles form a cluster digest.

In Zhao and Karypis (2005) constrained agglomerative algorithms are introduced. In this approach, a partitional clustering algorithm is used to compute a k-way clustering solution. Then, each of these clusters is treated as a separate collection and an agglomerative algorithm is used to build a tree for each of them.

Finally, the k different trees are merged by using an agglomerative algorithm that treats the documents of each subtree as an already formed cluster. Its time complexity is $O(n^{3/2}\log n)$ if k is reasonably large.

The two algorithms mentioned above are non-incremental methods. There are some incremental algorithms that update the cluster hierarchy when new documents arrive, such as Suffix Tree Clustering (Zamir, Etzioni, Madani, & Karp, 1997) and DC-tree (Wong & Fu, 2000).

The Suffix Tree Clustering (STC) algorithm does not treat a document as a set of words but rather as a string, making use of proximity information between words. STC relies on a suffix tree to identify sets of documents that share common phrases and uses this information to create clusters and to summarize their contents for users. However, the time complexity of STC is quite high with respect to the number of terms. The tree also suffers a great degree of redundancy.

DC-Tree is an incremental hierarchical algorithm for Web document clustering. Unlike the STC algorithm, DC-Tree is based on the vector–space representation of documents. Starting from the root, each new document descends the tree by iteratively choosing the most similar cluster. The process is repeated until a leaf node is reached, or the similarity between the document and the clusters at one level does not exceed a given threshold, in which case a new cluster node is created. The algorithm defines several parameters and thresholds, thus the parameter tuning is problematic. The main drawback of this algorithm is that document assignments to clusters are irrevocable.

As mentioned in the introduction, the topic detection problem also makes use of text clustering. The topics of a collection are unknown a priori and no previous knowledge about them is available. Recently, in the last topic detection and Tracking (TDT) competition, TDT 2004 (National Institute of Standards & Technology, 2004) hierarchical topic detection task was introduced, which is mainly intended to evaluate topic detection systems whose output can present clusters with different detail levels. More specifically, this task considers that the output of a TDT system consists of a lattice of clusters, where the top is the entire collection and the bottom comprises the singleton clusters. For evaluating these systems, it has been proposed several measures that combine both normalized detection cost and a lattice travelling cost. However, the set of manually identified topics employed to evaluate them is not hierarchically organized, although it contains topics with different detail levels. Since we mainly concern with the inner structure of topics, regardless its detail level, this method cannot be properly applied to evaluate our system.

ICT (Yu et al., 2004) and UMASS (Cornell et al., 2004) are two algorithms that have been proposed for the hierarchical topic detection tasks. ICT batches the documents (stories) in many buckets by time order. It firstly uses an agglomerative hierarchical clustering method in each bucket to produce micro-clusters. Finally, the hierarchy is built, by applying successively the Single-Pass clustering. UMASS system proposes two steps, bounded 1-NN for event formation and bounded agglomerative clustering for building the hierarchy. The algorithm ends when the number of clusters is smaller than a given value. The generated clusters by both methods depend on the document arrival order and they do not provide any summary for the detected topics.

To sum up, we compare the hierarchical clustering algorithms mentioned above considering if they are incremental, if they provide mechanisms to obtain summaries, if they depend on the document arrival order, and finally its complexity. As Table 1 shows, our method is incremental, order-independent and provides summaries of the clusters. Its time complexity is linear if the number of hierarchy levels is fixed, otherwise it is subquadratic.

Table 1
Comparison of hierarchical algorithms

Algorithm	Incremental	Summaries	Time complexity	Order dependency
Scatter/Gather	No	Topical words and typical titles	$O(kn\log n)$	Yes
Constrained agglomerative algorithms	No	No	$O(n^{3/2}\log n)$	Yes
STC	Yes	Common phrases	Linear, but space demanding	No
DC-Tree	Yes	No	Subquadratic	Yes
ICT	No	No	Subquadratic	Yes
UMASS	No	No	Subquadratic	Yes
Our method	Yes	Extracted sentences	Linear	No

3. Global architecture

The global architecture of our method is presented in Fig. 2. Starting from a collection of documents, the system first processes them to get a proper representation in order to cluster them accordingly to their semantics. The hierarchical clustering algorithm is then applied to obtain the structure of topics and subtopics of the collection. Finally, the summarization method is applied to obtain the summaries of the discovered topics.

3.1. Document processing

The steps included in this pre-processing are the following ones: document texts are tokenised, abbreviations and proper names are recognised, stop-words are removed using a standard stop-word list, part-of-speech tags are added and common words are replaced by their lemmas. As a tool for annotating text with part-of-speech and lemma information we used the Tree-Tagger parser, developed by the University of Stuttgart. All these elements are called terms. Additionally, we also extract some useful meta-data such as the publication dates and the places mentioned in the texts.

3.2. Document representation

From a journalist's perspective, a news story describing a topic will typically specify the following information: when it occurred, who was involved, where it took place and how it happened. Therefore, we think that these properties are very useful to describe and summarize the topics.

Most approaches do not consider these properties, representing the whole document as a bag of textual terms. As a consequence, with this representation it is hard to detect for example whether two different train accidents are two different topics for terms occurring in the involved documents are very similar. There are some systems such as Kumaran and Allan (2004) that use named entities to create three vector representations for each document (all terms, only named entities and non-named entity terms). However, the score of each vector depends on the news category (e.g. elections, accidents, etc.). Therefore, they need a training sample to determine the simple rules for each category. Stokes and Carthy (2001) define a composite document representation based on the lexical chains derived from a text and traditional keyword index terms. Makkonen, Ahonen-Myka, and Salmenkivi (2004) has recently presented a topic detection approach that employs semantic classes in the representation of documents. They split the term space into four semantic classes: places, names, temporal expressions and general terms, and they use class-wise similarity measures. Since these

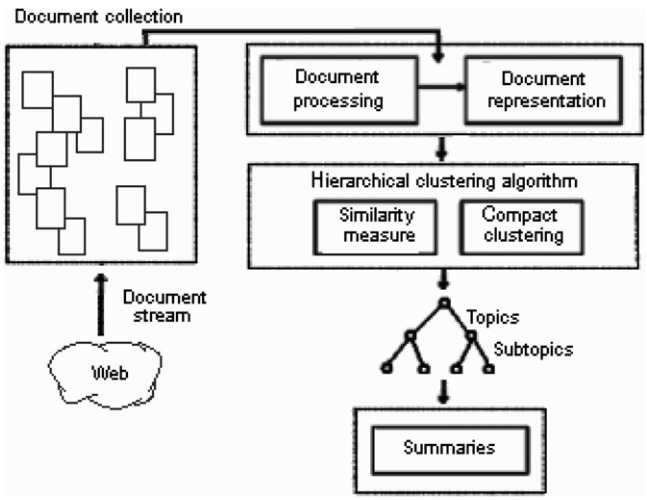

Fig. 2. Global architecture of our method.

sub-vectors reside in different representation spaces, it is crucial and difficult to determine the relative emphasis of each semantic class.

In this paper, we propose a specific representation for each semantic property of news: the actions and their main protagonists (what and who?), their time properties (when?) and their space properties (where?). The main aim of this representation is not only to compare news taking into account these properties, but also to provide a useful way of describing and summarizing the topics. Thus, our system builds three feature vectors to represent each document d^i, namely:

- *A vector of weighted terms* $T^i = (w_1^i, \ldots, w_n^i)$, where terms represent the lemmas of the words appearing in the content of the document, n is the total number of terms and w_k^i is the SMART *ltc* weight (Buckley, Salton, Allan, & Singhal, 1995) of the term t_k in d^i.
- *A vector of publication dates.* A document only contains its publication date, whereas cluster representatives have a selection of publication dates taken from their members. In our previous work (Pons-Porrata, Berlanga-Llavori, & Ruiz-Shulcloper, 2002b) we use a *vector of weighted time entities instead a vector of publication dates.* These time entities are either dates or date intervals that are automatically extracted from texts by using the algorithm presented in (Llidó, Berlanga, & Aramburu, 2001). Earlier experiments showed us that using a vector of publication dates not only reduces the computational cost but also produces better results for our clustering algorithm.
- *A vector of weighted places* $(TF_{p_1^i}, \ldots, TF_{p_{l_i}^i})$, where $TF_{p_k^i}$ is the absolute frequency of the place p_k in d^i and $k = 1, \ldots, l_i$. Place names are automatically detected from the document's contents by using a thesaurus built from the National Imagery and Mapping Agency files (NIMA).[1] Internally, each place name is represented as a path of codes with the different geographic regions where the place is included (country, administrative region, etc.). For ambiguous place names, the detection algorithm selects those paths involving the most frequently mentioned countries in the document.

3.3. Document similarity measure

Automatic document clustering is based on a similarity measure and a clustering criterion. The widest adopted document similarity measure has been the well-known cosine measure. In our case, we also introduce three similarity comparison criteria, each one corresponding to the components of the proposed document representation.

To compare the term vectors of two documents d^i and d^j we use the traditional cosine measure:

$$S_T(d^i, d^j) = \cos(T^i, T^j) = \frac{\sum_{k=1}^{n} w_k^i \cdot w_k^j}{\sqrt{\sum_{k=1}^{n} (w_k^i)^2} \sqrt{\sum_{k=1}^{n} (w_k^j)^2}}$$

To compare the time vectors of two documents we propose the following function: $D(d^i, d^j) = \min_{f^i \in FR^i, f^j \in FR^j} \{d(f^i, f^j)\}$, where $d(f^i, f^j)$ is the number of days between both and FR^i is the set of all publication dates f^i of the document d^i. This function is based on the traditional distance between sets.

To compare the place vectors of two documents we propose the following Boolean function:

$$S_p(d^i, d^j) = \begin{cases} 1 & \text{if } \exists p_q^i, p_i^j \text{ such that they have a common prefix or } d^i \text{ or } d^j \text{ has no place references} \\ 0 & \text{otherwise} \end{cases}$$

For example, "Kosovo" represented by 5G.02, is similar to "Yugoslavia" represented by 5G, because both have a common prefix. As it can be noticed, we consider that two documents are similar with respect to its place vectors if both contain at least one place belonging to the same country. However, if one of the documents have no place references in their contents (absence of information), they are considered similar (rather not dissimilar).

[1] ftp://ftp.nima.mil/pub/gns_data.

Finally, the global similarity measure can be defined as follows:

$$S(d^i, d^j) = \begin{cases} S_T(d^i, d^j) & \text{if } S_p(d^i, d^j) = 1 \wedge D(d^i, d^j) \leqslant \beta_{\text{time}} \\ 0 & \text{otherwise} \end{cases}$$

where β_{time} is the maximum number of days that are required to determine whether two articles refer to the same or to different topics. This measure is intended to capture the idea that two documents reporting a same topic should have a high semantic similarity, time proximity and place coincidence. Moreover, time and space components are used as filters in the global measure, which can reduce notably the complexity of the clustering algorithm.

It worth mentioning that more complex similarity measures for time and space components can be defined. However, our previous experiments demonstrated that this simpler measure works as well as the complex ones for our clustering algorithm (Pons-Porrata, Berlanga-Llavori, & Ruiz-Shulcloper, 2002a).

4. Document clustering algorithm

Traditional hierarchical clustering algorithms (e.g. complete-link, single-link, average-link, etc.) are intended to build a hierarchy of document classes. However, the levels of the generated hierarchies correspond to the steps of the clustering algorithm (Jain & Dubes, 1988), but not always to the required abstraction levels of our application. Instead of that, in our hierarchical clustering algorithm each level of the hierarchy consists of a set of clusters of abstract documents, which summarize those generated in the previous level.

In the following section we describe how cluster representatives are obtained, and afterwards we present the proposed clustering algorithm.

4.1. Cluster representatives

The representative of a cluster c, denoted as $\bar{c}$, is a tuple $(T^{\bar{c}}, F^{\bar{c}}, P^{\bar{c}})$, of terms, temporal and place components, respectively. In this work, it is calculated as the union of the cluster's documents:

$T^{\bar{c}} = (T^{\bar{c}}_l, \ldots, T^{\bar{c}}_n)$, where $T^{\bar{c}}_j$ is the weight average of term t_j in the cluster's documents.
$F^{\bar{c}} = (F^{\bar{c}}_{f_1}, \ldots, F^{\bar{c}}_{f_s})$, where $F^{\bar{c}}_{f_j}$ is the absolute frequency of the date f_j in the cluster, that is, the number of documents in the cluster containing this date, and s is the total number of publication dates that describe the documents of this cluster.
$P^{\bar{c}} = (P^{\bar{c}}_{p1}, \ldots, P^{\bar{c}}_{pl})$, where $P^{\bar{c}}_{p_j}$ is the absolute frequency of place p_j in the cluster, and l is the total number of places mentioned in the documents of this cluster.

In order to reduce the length of the cluster representatives, their vectors are truncated by removing the terms (dates, places) whose frequency is smaller than the 10th part of the vector maximum frequency.

4.2. Incremental hierarchical clustering algorithm

Similarly to constrained agglomerative algorithms (Zhao & Karypis, 2005), our hierarchical algorithm combines features from both partitional and agglomerative approaches. The proposed algorithm aims at building a topic hierarchy by clustering successively the representatives of the clusters obtained in the previous hierarchy level. In this way, the higher the hierarchy level is, the less details in the identified topics are obtained. The lowest level of the hierarchy is intended to allocate the smallest identifiable topics, which are groups of tightly related documents. The rest of the levels contain cluster representatives.

The incremental hierarchical algorithm we propose (see Fig. 3) uses a clustering routine that must fulfill the following requirements:

- It must be incremental.
- It should not depend on the order of the incoming documents.

Input: Similarity measure *S* and its parameters; Clustering routine and its parameters.
Output: Hierarchy of document clusters representing the identified topics.
Step 1. Arrival of a document *d* and *Level* = 1.
Step 2. Apply the clustering routine in the first level of the cluster hierarchy.
Step 3. Let *RS* be the set of clusters that are removed when applying the clustering routine.
Let *NS* be the set of the new clusters that are formed.
Increment *Level*.
Let *Q* be a queue with the cluster representatives to be processed in this level, Q = ∅.
Step 4. Remove all the representatives of the clusters belonging to *RS* in *Level*.
Put into the queue *Q* all members of the clusters in *Level* where at least one representative was eliminated. Remove these clusters from the list of the existing clusters.
Step 5. Calculate the cluster representatives of *NS* and put them into the queue *Q*.
Step 6. For each cluster representative in the queue *Q* apply the clustering routine.
Step 7. If *Level* is the desired level of the hierarchy or all clusters in *Level* are singleton then Finish. Otherwise, Go to the Step 3.

Fig. 3. Incremental hierarchical clustering algorithm.

- It must use a similarity measure that takes into account the temporal-spatial proximity of documents and representatives.

Since the proposed hierarchical algorithm is incremental, it is assumed that it deals with an existing hierarchy of documents (initially empty), where the first level consists of a partition of the document collection, and the next levels contain the cluster representatives of each previous level. When a new document arrives at the system, the clusters in all levels of the hierarchy must be revised. Firstly, the algorithm applies a clustering routine in the first level of the hierarchy in order to incorporate the new document to the clustering. This process can create new clusters and remove existing ones.

When clusters are removed from a level of the hierarchy, their representatives must be removed from the clusters in which they were located in the next level. Similarly, when new clusters are created, their representatives must be calculated. The members of the clusters where changes took place (that is, some cluster representatives were eliminated) as well as all new representatives, must be queued in order to incorporate them to the set of existing clusters in this level of the hierarchy. For that purpose, we apply the clustering routine again. This process repeats until the desired level of the hierarchy is reached or all clusters are singleton. It is worth mentioning that in each level of the hierarchy different parameters of the similarity measure and of the clustering routine, and even different clustering routines can be used.

In Fig. 4(a) an example of a hierarchy of clusters is shown. The first level of the hierarchy contains 10 clusters, whereas the second one contains four clusters. As it can be noticed, a cluster in the second level is formed

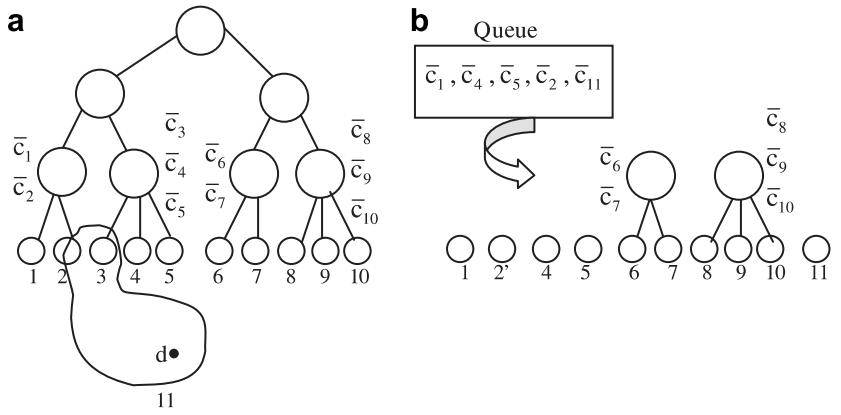

Fig. 4. A hierarchy of clusters.

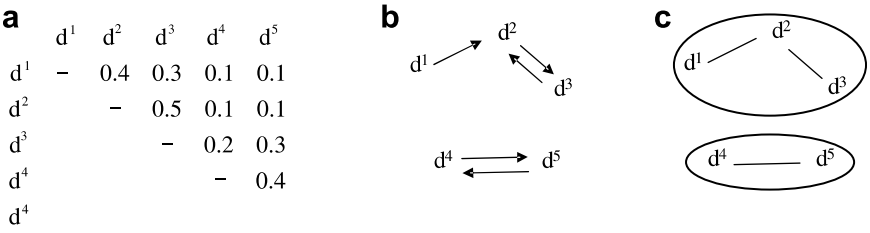

Fig. 5. β_0-Maximum similarity graph and β_0-compact sets with $\beta_0 = 0.3$: (a) similarity matrix, (b) β_0-maximum similarity graph and (c) β_0-compact sets.

by the representatives of the clusters 3, 4 and 5 of the first level. When the new document d arrives, a new cluster (11) is created with some documents of the cluster 2 and all the documents of the cluster 3. The Fig. 4(b) shows the changes that take place in the second level of the hierarchy. Thus, the representatives $\bar{c}_2$ and $\bar{c}_3$ are removed and two new representatives $\bar{c}'_2$ and $\bar{c}_{11}$ are created. The representatives $\bar{c}_1, \bar{c}_4, \bar{c}_5, \bar{c}'_2$ and $\bar{c}_{11}$ are enqueued in order to incorporate them to the set of clusters in the second level of hierarchy. Notice that the clusters formed by the representatives $\bar{c}_6, \bar{c}_7, \bar{c}_8, \bar{c}_9$ and $\bar{c}_{10}$ do not change.

In this paper, we use the compact clustering algorithm as the basic clustering routine. It is based on the incremental construction of all β_0-compact sets. Basically, two documents are β_0-similar if its similarity is greater or equal than a user-defined threshold β_0. We call β_0-maximum similarity graph to the directed graph, whose vertices are the documents of the collection and there is an arc from the vertex d^i to the vertex d^j if d^j is the most β_0-similar document to d^i. The β_0-compact sets are equivalent to the connected components in the β_0-maximum similarity graph without taking into account the orientation of its arcs (see Fig. 5). More details of this incremental clustering algorithm and the mathematical properties in which is based can be seen in Pons-Porrata et al. (2002b).

The time complexity of the compact clustering algorithm is $O(n^2)$. However, if we use the proposed similarity measure, the complexity of the compact clustering algorithm is linear. The need for exhaustive pairwise comparisons of documents is avoided by using a time window. This window contains the documents in the time period defined by β_{time} days. Thus, each document is only compared with the documents in the time window, since the similarities with other documents are zero.

5. Cluster description method

As mentioned in Section 2, current clustering-based systems do not provide information enough so that users become aware of the cluster contents. In this section we propose a summarization method to provide informative cluster descriptions.

Basically, a multidocument summarization system tries to determine which sentences must be included in the summary, and then how to organize them to make the summary comprehensible. Many of these approaches are based on a sentence weight function that takes into account the position of the sentences in the documents, the length of the sentences, and the number of frequent keywords of the set of documents they include (Mani, 2001). In this way, all the sentences in the document cluster must be scored to select the most appropriate ones for the summary.

5.1. Summarization method

The temporal entities of a document cluster can be summarized as a date interval from the time vector of its cluster representative. Places can be easily summarized by taken a representative place from the cluster documents. Thus, in this section we only focus on the term vector to extract the cluster summaries.

One critical point in summary generation is the selection of "good" terms for extracting the most informative sentences. Current approaches usually take a list of terms from the (multi)document ranked by the TF-IDF weights. In these systems, IDF scores are taken from external sources, which can be not appropriate as they may not reflect properly the vocabulary frequencies of the collection at hand. On the other hand,

TF-IDF does not take into account interesting word co-occurrences containing terms with low IDF (i.e. they are frequent in the overall collection). For example, the term "hand" is used in many contexts, usually in set phrases (e.g. at hand, hand in hand, etc.), and therefore it presents a low IDF. However, in some news documents this word is very relevant to select the topic informative sentences, as for example in the TDT2 topic "Shoplifter's Hand Amputated". Similarly, some significant words of very large topics (e.g. "Monica Lewinsky Case") also have low IDF weights, and therefore they may be not selected for the summary.

In order to determine at a glance whether the contents of a topic are of user interest or not, we propose a new summarization method based on Testor Theory (Lazo-Cortés et al., 2001). Starting from a set of document clusters, each one representing a different topic, our method first selects the most frequent terms of each cluster that are not frequent in the other clusters. Then, it builds their co-occurrences that distinguish the cluster. Once these terms have been selected, the system extracts all the sentences containing the selected terms. In order to avoid redundant sentences in the summary, each selected sentence is compared with the others to detect high word overlapping. In this case, only the largest sentence is included in the summary. Finally, in order to improve the coherence and organization of the summaries, we sort the selected sentences according to the publication date of the news documents and their position in the text. The next section describes with detail the application of the Testor Theory for the selection of candidate terms.

5.2. Applying the Testor Theory for the selection of terms

In the Testor Theory, the set $\tau = \{x_{i_l}, \ldots, x_{i_k}\}$ of features and their corresponding set of columns $\{i_l, \ldots, i_k\}$ of a matrix M is called a *testor*, if after deleting from M all columns except $\{i_l, \ldots, i_k\}$, all rows of M corresponding to distinct classes are different. A testor is called *irreducible (typical)* if none of its proper subsets is a testor (Lazo-Cortés et al., 2001).

For each cluster c, we construct a matrix $MR(c)$, whose columns are the most frequent terms in the representative $\bar{c}$, and its rows are the representatives of all clusters, described in terms of these columns. In order to calculate the typical testors, we considered two classes in the matrix $MR(c)$. The first class is only formed by $\bar{c}$ and the second one is formed by the other cluster representatives. Notice that our goal is to distinguish the cluster c from the other clusters.

For the calculus of the typical testors of a matrix M, the key concept is the comparison criterion of the values of each feature. In our case, the features that describe the documents are the terms and its values are the frequency of terms. The comparison criterion applied to all the features is

$$d(v_{i_k}, v_{j_k}) = \begin{cases} 1 & \text{if } v_{i_k} - v_{j_k} \geqslant \delta \\ 0 & \text{otherwise} \end{cases}$$

where v_{i_k}, v_{j_k} are the frequencies in the cluster representative i and j in the column corresponding to the term t_k respectively, and δ is a user-defined parameter. As it can be noticed, this criterion considers the two values (frequencies of the term t_k) different if the term t_k is frequent in cluster i and not frequent in cluster j.

In order to determine all typical testors of a matrix we use the algorithm LEX (Santiesteban & Pons-Porrata, 2003), which computes efficiently the typical testors of a data collection.

A summary of a cluster c consists of a set of sentences extracted from the documents in c in which the highest quantity of terms that belong to the typical testors of the matrix $MR(c)$ occurs. Moreover, the sentences that cover the calculated typical testor set are also added to the summary. More details of this method can be seen in Pons-Porrata, Ruiz-Shulcloper, and Berlanga-Llavori (2003).

For example, let suppose that we have three clusters about (1) the economic sanctions on India's government for detonating three underground nuclear explosions, imposed by Bill Clinton, (2) the declarations of the president Clinton related to the situation of the human rights in China, and (3) the medallists in women's 500-meter short track speed skating in the Olympic games. We want to obtain the summary of the first cluster c_1. As mentioned before, we first obtain the most frequent terms in the representative of c_1. Suppose that these terms are "president_Bill_Clinton", "sanction", "United_States", "nuclear" "Indian" and "obtain". Then, we construct the matrix $MR(c_1)$ as follows:

	President Bill Clinton	Sanction	United States	Nuclear	Indian	Obtain
$\bar{c}_1$	0.12	0.09	0.08	0.15	0.11	0.07
$\bar{c}_2$	0.1	0	0.09	0	0	0.06
$\bar{c}_3$	0	0	0	0	0.09	0.08

Notice that most terms are not present in the representative of the cluster 3. From this matrix, we construct the difference matrix in order to calculate the typical testors, by applying the comparison criterion mentioned above. In this case, we use $\delta = 0.02$. Thus, the difference matrix is

	President Bill Clinton	Sanction	United States	nuclear	Indian	Obtain
$\bar{c}_1, \bar{c}_2$	0	1	0	1	1	0
$\bar{c}_1, \bar{c}_3$	1	1	1	1	0	0

The typical testors of this matrix are: {president_Bill_Clinton, Indian}, {sanction}, {nuclear} and {United_States, Indian}. All these terms are used to extract the sentences that will form the summary of c_1.

6. Experiments and results

The effectiveness of our method has been evaluated using the TDT2 dataset, version 4.0. This corpus consists of 6 months of news stories from January to June 1998. The news stories were collected from six different sources. Human annotators identified a total of 192 topics in the TDT2 dataset. Nine thousand eight hundred and twenty four English stories belong to one of these topics, the rest are unlabeled. This corpus is divided into three sub-collections, namely: training, development test and evaluation test, which are summarized in Table 2. Unfortunately the TDT collections are not annotated at different abstraction levels, and therefore, we cannot evaluate properly the composition of topics and subtopics. Instead, we evaluate each level of the generated hierarchy against all the TDT2 topics to observe their impact in different tasks: topic detection and topic retrieval.

We use two quality measures of the literature that compare the system-generated clusters with the manually labeled topics, namely: the $F1$-measure and the detection cost (National Institute of Standards & Technology, 1998). The $F1$-measure is widely applied in Information Retrieval systems, and it combines the precision and recall factors. In our case, the $F1$-measure of the system-generated cluster number j with respect to the manually labeled topic number i can be evaluated as follows:

$$F1(i,j) = 2 \cdot \frac{n_{ij}}{n_i + n_j}$$

where n_{ij} is the number of common members in the topic i and the cluster j, n_i is the cardinality of the topic i, and n_j is the cardinality of the cluster j.

To define a global measure, first each event must be mapped to the cluster that produces the maximum $F1$-measure:

$$\sigma(i) \arg\max_j \{F1(i,j)\}$$

Table 2
Description of the TDT2 collections

Collection	Number of topics	Date range	Number of documents	Number of terms
Evaluation	34	4/1–30/6	1781	17,678
Development	26	4/1–29/6	905	11,444
Training	36	4/1–30/6	5409	35,882
Global	192	4/1–30/6	9824	55,112

Hence, the macro $F1$-measure is calculated as follows:

$$F1 = \frac{1}{N_{\text{topics}}} \sum_{i=1}^{N_{\text{topics}}} F1(i, \sigma(i))$$

The detection cost is a measure that combines both the miss and false alarm errors between a topic i and a system-generated cluster j:

$$C_{\text{DET}}(i,j) = C_{\text{miss}} \cdot P_{\text{miss}}(i,j) \cdot P_{\text{topic}} + C_{\text{false\_alarm}} \cdot P_{\text{false\_alarm}}(i,j) \cdot (1 - P_{topic})$$

where $P_{\text{miss}} = (n_i - n_{ij})/n_i$ and $P_{\text{false\_alarm}} = (n_j - n_{ij})/(N - n_i)$ are the conditional probabilities of a miss and a false alarm, respectively, P_{topic} is the a priori probability of a document belonging to a given topic, C_{miss} and $C_{\text{false\_alarm}}$ are the costs of a miss and a false alarm, respectively, and N is the collection size. Following a convention in the TDT evaluations, we assign $C_{\text{false\_alarm}} = 0.1$, $C_{\text{miss}} = 1$ and $P_{\text{topic}} = 0.02$.

Again, to define the final measure, each topic must be mapped to the cluster that produces the minimum detection cost:

$$\sigma(i) = \arg\min_j \{C_{\text{DET}}(i,j)\}$$

The macro-average of this measure (also called *Topic Weighted*) is then defined as follows:

$$C_{\text{DET}} = \frac{1}{N_{\text{topics}}} \sum_{i=1}^{N_{\text{topics}}} C_{\text{DET}}(i, \sigma(i))$$

Finally, the normalized detection cost is calculated as follows:

$$(C_{\text{DET}})_{\text{Norm}} = \frac{C_{\text{DET}}}{\min\{C_{\text{m}} \cdot P_{\text{topic}}, C_{\text{fa}} \cdot (1 - P_{\text{topic}})\}}$$

Table 3 shows the maximum $F1$-measure obtained by our method on the TDT2 dataset and its sub-collections. The first row shows results for our method using only the cosine measure, whereas the second and third rows show results when using our similarity measure instead. The second row only considers terms and the time component, whereas the third one considers terms, time and space components. Each cell contains the $F1$ values considering a hierarchy of one level, two levels and so on. For the evaluation, we consider the flat partition produced by each level of the hierarchy and we compare this partition with the manually identified topics. The entries that are boldfaced correspond to the level that performed the best in each document collection.

Table 3
$F1$-Measure for the TDT2 collections

Measure	Evaluation	Development	Training	Global
Base cosine	0.5735	0.6056	0.4596	0.5424
	0.8295	0.861	0.7922	**0.7892**
	0.9233	**0.9469**	0.8369	0.7806
	0.9294		0.854	
			0.8599	
Base cosine + time filter	0.5402	0.5874	0.436	0.5462
	0.8077	0.8799	0.76	**0.8129**
	0.8322	0.9242	0.8429	0.7595
	0.7968	0.9283	**0.8583**	
		0.9284		
Base cosine + spatio-temporal filter	0.539	0.5824	0.4385	0.5403
	0.7738	0.8645	0.7684	**0.7955**
	0.8629	0.9182	0.8449	0.7728
	0.8738	**0.9219**	**0.8634**	

In our experiments we use the same β_0 value in all levels of the hierarchy, but β_{time} is empirically fixed to 20 and 50 days in the bottom and the remaining levels, respectively. To statistically compare the obtained results we used the paired t-test (Devore & Peck, 1997), with 95% of confidence.

Several observations can be made by analyzing the results in Table 3. First, the performance difference between the cosine measure and our similarity function is not statistically significant. Also, the performance difference between using only the temporal component and the spatio-temporal one is not statistically significant either. Taking into account that the system effectiveness does not diminish when introducing the place filter, we can conclude that extracted places are valid to describe the topics. In general, regarding time and spatial components do not affect the quality of topics. Therefore, they can be applied without losing effectiveness to reduce the time complexity of the compact clustering algorithm from $O(n^2)$ to $O(n)$, being n the number of documents (Pons-Porrata et al., 2002b).

Second, the difference between the one-level and two-level results is extremely statistically significant. Thus, results in two-level hierarchies are much better than those obtained by one-level hierarchies in all the evaluated collections, which demonstrates that hierarchical clustering properly captures the different granularities of the TDT2 topics. Moreover, $F1$ values in smaller document collections still improve if more levels of the hierarchy are added. It is worth mentioning that $F1$ values are very good, being the proposed system a good tool for retrieval purposes. Moreover, even being an unsupervised system, its results are comparable with supervised classification systems.

With regard to the detection cost, Table 4 shows the obtained results following the same structure. This table includes the values of topic weighted detection cost and normalized detection cost obtained in each collection. In this case, the difference between cosine measure and our similarity measure is not quite statistically significant according to the paired t-test.

Looking at the results of one-level and two-level hierarchies, we can see that they are in agreement with the results presented earlier in Table 3. We consider that the detection cost values obtained by our method are good enough if we compare them with those obtained by the existing detection systems (see evaluation results in Fiscus, Doddington, Garofolo, & Martin, 1999).

Table 5 shows the best quality results obtained for each document collection using the base cosine with the time filter (results obtained with base cosine with the spatio-temporal filter are very similar). As we can see, the precision values are quite high (superior to 0.97) in the first level of all hierarchies, but the recall values are low. The higher the level, the better recall values are obtained, while the precision slightly decreases. Also, the best values of detection cost are obtained in the third level of the hierarchy in most cases. Thus, we think that the first level can be associated to event-retrieval tasks due to the high precision obtained. The second level of the hierarchy allows users to retrieve topics, and the third level can be used for topic detection tasks.

Table 4
Detection cost for the TDT2 collections

Measure	Evaluation		Development		Training		Global	
	Cost	Normalized cost	Cost	Normalized cost	Cost	Normalized cost	Cost	Normalized cost
Base cosine	0.0105	0.5278	0.0099	0.4973	0.0130	0.6505	0.0105	0.5288
	0.0044	0.2219	0.0036	0.1803	0.0048	0.2429	0.0028	0.1424
	0.0020	0.1034	**0.0011**	**0.0563**	0.0028	0.1400	**0.0021**	**0.1063**
	0.0019	**0.0969**			0.0027	0.1360	0.0042	0.2136
					0.0026	**0.1307**		
Base cosine + time filter	0.0115	0.5753	0.0110	0.5503	0.0133	0.6679	0.0110	0.5549
	0.0045	0.229	0.0039	0.1968	0.0058	0.2935	0.0036	0.1807
	0.0028	**0.1426**	0.0020	0.1019	**0.0030**	**0.1544**	**0.0021**	**0.1089**
	0.0030	0.1516	0.0018	0.0946	0.0031	0.1578	0.0045	0.225
			0.0018	**0.0945**				
Base cosine + spatio-temporal filter	0.0115	0.5762	0.011	0.5556	0.0133	0.6655	0.0118	0.5901
	0.0056	0.2788	0.0044	0.2225	0.0056	0.2819	0.0050	0.2334
	0.0032	0.1625	0.0026	0.1289	**0.0031**	**0.1551**	**0.0031**	**0.1540**
	0.0029	**0.1470**	**0.0024**	**0.1219**	0.0031	0.1553		

Table 5
Results for the best hierarchies obtained in the TDT2 collections

Collection	$F1$	Precision	Recall	Cost	Normalized cost	P_{fa}	P_{MISS}
Evaluation	0.5402	0.9926	0.4249	0.0115	0.5753	6.04e−5	0.5750
	0.8077	0.9452	0.7759	0.0045	0.2290	0.0015	0.2215
	0.8322	0.8787	0.8866	**0.0028**	**0.1426**	0.0059	0.1133
Development	0.5874	0.9904	0.4742	0.0110	0.5503	0.0001	0.5495
	0.8799	0.9892	0.8285	0.0039	0.1968	0.0003	0.1952
	0.9242	0.9444	0.9558	0.0020	0.1019	0.0069	0.0679
	0.9283	0.9444	0.9633	0.0018	0.0946	0.0069	0.0604
	0.9284	0.9444	0.9635	**0.0018**	**0.0945**	0.0069	0.0602
Training	0.436	0.9781	0.3323	0.0133	0.6679	4.4e−5	0.6676
	0.76	0.9503	0.7094	0.0058	0.2935	0.0006	0.2905
	0.8429	0.8905	0.8877	**0.0030**	**0.1544**	0.0056	0.1269
	0.8583	0.8990	0.9250	0.0031	0.1578	0.0077	0.1199
Global	0.5462	0.9789	0.4225	0.0116	0.5827	2.7e−5	0.5825
	0.8129	0.9221	0.7924	0.0042	0.2112	0.0003	0.2095
	0.7595	0.7808	0.8978	**0.0024**	**0.1231**	0.0040	0.1033

Table 6 shows the $F1$ results for 20 TDT target topics randomly selected. Last column contains the hierarchy level at which the best $F1$ value is obtained for each topic. As it can be noticed, our algorithm identifies topics accordingly to its specificity. Thus, the broader a topic is, the higher is the level at which is detected.

Fig. 6 shows a snippet of the topic hierarchy created by our method for the TDT2 collection. This hierarchy is represented as an XML file containing for each topic its summary and the subtopics it comprises. The summaries capture the main ideas about each topic and an appreciable reduction of words is also achieved (superior to 70% in most of the topics). The whole hierarchy can be seen in http://krono.act.uji.es/Experiments.

Our last set of experiments was focused on evaluating the impact of typical testors in the construction of summaries. In the experiments we want to demonstrate if the terms selected with typical testors are representative enough of the topics they describe. With this purpose, we conducted an experiment where all documents

Table 6
Results for 20 TDT target topics

Topic	Number of documents	$F1$	Level
Asian economic crisis	1133	0.7417	4
Current conflict with Iraq	1486	0.9322	4
Monica Lewinsky case	969	0.9310	4
India – a nuclear power?	475	0.9542	3
Bombing AL clinic	99	0.9949	3
Pope visits Cuba	151	0.9730	3
1998 Winter Olympics	540	0.8608	5
Cable car crash	110	0.9909	4
Shevardnadze assassination attempt	38	1.0	3
Mountain hikers lost	7	1.0	1
Capps replacement elections	5	1.0	2
Spanish dam broken	7	1.0	2
India parliamentary elections	121	0.9304	3
GM strike	142	1.0	3
Rats in space!	60	0.8349	2
Race relations meetings	12	0.9091	3
Mandela visits Angola	2	1.0	1
Tornado in Florida	53	0.9811	2
World AIDS Conference	21	0.9756	2
Israeli–Palestinian Talks (London)	210	0.8668	3

```
- <Topic Id= "Cluster_2775" Level= "1">
    <Description Place="Afghanistan" Date="[19980530-19980606]"  Summary= "A United Nations relief team
     is on its way to a remote area of northern Afghanistan, which was struck by a powerful earthquake. In the
     village of Kol, hundreds of people swarmed a United Nations helicopter that touched down three days after
     Saturday's earthquake struck a remote mountainous area rocked three months earlier by another massive
     quake that claimed some 2,300 victims."  Testors="[remote,afghanistan,relief,helicopter,village,quake,area,
     northern,earthquake]"/>
   - <Event Id= "Cluster_8946" Level= "1">
     <Description Place="Afghanistan" Date="[19980602-19980604]"  Summary= "Aid agencies in northern
     Afghanistan say Saturday's earthquake has damaged a much wider area than first thought. Aid workers say
     more than 4,000 people may have died and survivors have spoken of a desperate need for food. Heavy after
     shocks from an earthquake that killed thousands of people in Afghanistan continue to damage more villages
     and add to the misery of survivors. They reckon they surveyed 50 villages either by actually dropping down
     and going into the villages themselves or just by flying over. Food and supplies are continuing to be
     distributed but more helicopters are needed to reach the victims. Aid workers are hoping too large
     helicopters donated by the United States will reach the region within the next few days. U.N. officials are also
     trying to bring more fuel into the region to increase the time the helicopters can stay in the air. U.N.
     spokeswoman Sarah Russell says more international assistance is needed to boost rescue and relief
     efforts."  Testors="[earthquake,reckon,damage,spokeswoman,food,shock,survivor,aid,rescue,helicopter,u.n.,
     village, Afghanistan,sara_russell]"/>
       <Document  SCode="VOA19980604.1800.1830"></Document>
       <Document SCode="PRI19980602.2000.1929 "></Document>
     </Event>
   - <Event Id= "Cluster_8878" Level= "1">
     <Description Place="Afghanistan" Date="[19980602-19980603]"  Summary= "…"  Testors="[ Afghanistan,
      bodoxshon_provinces,devastating,scour,reach,hillside,bearing,victim,helicopter,northern,earthquake,Takar,
      Talabon"/>
       <Document SCode="VOA19980603.1800.0238"></Document>
       <Document SCode="VOA19980603.1700.2078 "></Document>
```


Fig. 6. A snippet of the created hierarchy in TDT2 collection.

Table 7
Impact of typical testors

Collection	Number of terms	$F1$-measure	Cost	Normalized cost
Evaluation	2949 (83.3%)	0.8141 (+0.8%)	0.0044 (+2%)	0.2236 (+2.4%)
Development	3262 (71.5%)	0.8733 (−0.7%)	0.0043 (−10.2%)	0.2158 (−9.6%)
Training	9083 (74.7%)	0.729 (−4%)	0.0065 (−12%)	0.3272 (−11.5%)
Global	18322 (66.8%)	0.7962 (−2%)	0.0039 (−8.3%)	0.1952 (−8%)

of the collection are described only with terms belonging to the obtained typical testors of the topics in hierarchies of two levels. In other words, we use the typical testors as a feature selection technique. Once document representation is reduced to those terms, we perform again all the hierarchical clustering process. If the selected terms are representative enough, then the obtained clusters should present a similar or better quality with respect to using the original document representation.

Table 7 shows the reduction in the representation space when considering only typical testor terms, and the best values for detection cost and $F1$ using the reduced representation space. As it can be noticed, the dimensionality reduction is considerable whereas differences in the quality measures are not statistically significant according to the paired t-test.

7. Conclusions

In this paper we propose a new hierarchical clustering algorithm that combines partitional and agglomerative approaches to produce topic hierarchies. This algorithm obtains cohesive clusters with arbitrary shapes and it does not require fixing a priori the number of clusters. Another advantage of this algorithm is that it is incremental, being suited for web-based applications, where continuously new documents arrive. Moreover,

the generated set of clusters is unique and independent of the document arrival order, making it insensitive to the particular way documents arrive to the system.

The proposed similarity measure between documents considers their place names, time references and textual terms. Such a measure does not always improve quality results but it reduces the time complexity of the clustering algorithm from quadratic to linear.

Additionally, we propose a multidocument summarization method. It employs the calculus of typical testors as its primary operation and from them it constructs the summaries of each cluster. The most important novelty is the use of typical testors combined with different techniques and heuristics, to produce all together better summaries.

These Text Mining techniques are combined together to build a topic discovery system. The hierarchy obtained by our clustering algorithm allows users to discover the structure of topics and subtopics of a document collection, that is, to identify not only the topics but also the possible smaller subtopics they comprise. The multidocument summarization method provides a summary for each system-generated topic at any level of the hierarchy. This method is helpful to a user in order to determine at a glance whether a topic is of its interest. This approach avoids the definition of what a topic is, since is the user who decides in which level she explores the knowledge of interest. The method is particularly helpful in situations in which it is difficult or undesirable to specify a query formally.

Experiments demonstrate the validity of our algorithms for both Information Retrieval and Topic Detection tasks. Our system not only obtains similar results to the best systems in TDT2 evaluation but it also offers additional information to users: the hierarchy of topics and subtopics along with their description. Retrieval quality is also very good. We have also realized that each hierarchy level can be used for different user tasks. Thus, the first level can be associated to event-retrieval tasks, the second one to topic retrieval and the third level to topic detection tasks.

We believed that the accuracy of our method could be improved by optimizing the parameter values (β_0 and β_{time}). This provides further motivation to study in depth this problem. Another line of future work will be the study of new ways of introducing the spatial information in the global similarity measure to improve the clustering and summary qualities. Also, one key issue we face is the use of overlapped clustering routines in our hierarchical algorithm instead the compact clustering routine to allow for multitopic documents. Future work also includes enhancing the summary coherence by using, for example, anaphora resolution techniques.

References

Allan, J., Carbonell, J., Doddington, G., Yamron, J., & Yang, Y. (1998). Topic detection and tracking pilot study: final report. In *Proceedings of DARPA broadcast news transcription and understanding workshop, Lansdowne, VA* (pp. 194–218).

Barzilay, R., Elhadad, N., & McKeown, K. (2002). Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research, 17*, 35–55.

Buckley, C., Salton, G., Allan, J., & Singhal, A. (1995). Automatic query expansion using SMART: TREC-3. In *Proceedings of the third text retrieval conference (TREC-3)* (pp. 69–80). NIST Special Publication 500-225, Washington.

Cornell, M., Feng, A., Kumaran, G., Raghavan, H., Shah, C., & Allan, J. (2004). UMASS at TDT 2004. In *Proceedings of TDT 2004 workshop, Gaithersburg, MD*.

Cutting, D. R., Pedersen, J. O., Karger, D. R., & Tukey, J. W. (1992). Scatter/Gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th international ACM SIGIR Conference on research and development in information retrieval, Copenhagen, Denmark* (pp. 318–329).

Devore, J., & Peck, R. (1997). *Statistics: The exploration and analysis of data*. Belmont, CA: Duxbury Press.

Feldman, R., & Dagan, I. (1995). Knowledge discovery in textual databases (kdt). In *Proceedings of the first international conference on knowledge discovery and data mining (KDD-95), Montreal, Canada* (pp. 112–117).

Fiscus, J., Doddington, G., Garofolo, J., & Martin, A. (1999). Nist's 1998 topic detection and tracking evaluation (tdt2). In *Proceedings of the DARPA broadcast news transcription and understanding workshop, Herndon, VA* (pp. 19–24).

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall.

Kumaran, G., & Allan, J. (2004) Text classification and named entities for new event detection. In *Proceedings of the SIGIR conference on research and development in information retrieval SIGIR 2004, Sheffield, UK* (pp. 297–304).

Lazo-Cortés, M., Ruiz-Shulcloper, J., & Alba-Cabrera, E. (2001). An overview of the concept testor. *Pattern Recognition, 34*(4), 13–21.

Llidó, D., Berlanga, R., & Aramburu, M. J. (2001). Extracting temporal references to automatically assign document event-time periods. In *Proceedings of database and expert system applications* (pp. 62–71). Munich: Springer-Verlag.

Makkonen, J., Ahonen-Myka, H., & Salmenkivi, M. (2004). Simple semantics in topic detection and tracking. *Information Retrieval, 7*, 347–368.

Mani, I. (2001). *Automatic summarisation*. John Benjamins Publishing Company.

Mani, I., & Bloedorn, E. (1997). Multi-document summarization by graph search and matching. In *Proceedings of the fourteenth national conference on artificial intelligence (AAAI-97), Providence, RI* (pp. 622–628).

National Institute of Standards and Technology (1998). The topic detection and tracking phase 2 (TDT2) evaluation plan, version 3.7. Available from http://www.nist.gov/speech/tests/tdt/index.htm.

National Institute of Standards and Technology (2004). The 2004 Topic detection and tracking (TDT 2004) task definition and evaluation plan, version 1.0, 2004. Available from http://www.nist.gov/speech/tests/tdt/index.htm.

Pons-Porrata, A., Berlanga-Llavori, R., & Ruiz-Shulcloper, J. (2002a). Temporal-semantic clustering of newspaper articles for event detection. In *Pattern Recognition in Information Systems* (pp. 104–111). ICEIS Press.

Pons-Porrata, A., Berlanga-Llavori, R., & Ruiz-Shulcloper, J. (2002b). On-line event and topic detection by using the compact sets clustering. *Journal of Intelligent and Fuzzy Systems, 12*(3–4), 185–194.

Pons-Porrata, A., Ruiz-Shulcloper, J., & Berlanga-Llavori, R. (2003). A method for the automatic summarization of topic-based clusters of documents. *Lecture Notes on Computer Science, 2905*, 596–603.

Santiesteban, Y., & Pons-Porrata, A. (2003). LEX: a new algorithm for the calculus of typical testors. *Mathematics Sciences Journal, 21*(1), 85–95.

Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. InKDD workshop on text mining, Boston, MA.

Stokes, N., & Carthy, J. (2001). First story detection using a composite document representation. In *Proceedings of human language technology conference (HLT-01), San Diego, CA* (pp. 134–141).

Wong, W. C., & Fu, A. (2000). Incremental document clustering for web page classification. In *IEEE 2000 international conference on information society in the 21st century: Emerging technologies and new challenges (ISI2000), Japan*.

Yu, Man-Quan, Luo, Wei-Hua, Zhou, Zhao-Tao, et al. (2004). ICT's approaches to HTD and tracking at TDT2004. In *Proceedings of TDT 2004 workshop, Gaithersburg, MD*.

Zamir, O., Etzioni, O., Madani, O., & Karp, R.M. (1997). Fast and intituitive clustering of web documents. In *Proceedings of KDD '97, Newport Beach, CA* (pp. 287–290).

Zhao, Y., & Karypis, G. (2004). Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning, 55*(3), 311–331.

Zhao, Y., & Karypis, G. (2005). Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery, 10*, 141–168.

Aurora Pons-Porrata is a professor in the Computer Science career at Universidad de Oriente, Cuba for 14 years. She received the B.S. in Mathematics and M.S. degrees in Computer Science both from this university, and the Ph.D. degree in Computer Science from Universitat Jaume I, Spain in 2004. She is author of several articles in journals and proceedings, such as Intelligent and Fuzzy Systems and Mathematical Computer and Modelling. She is founder and member of the executive board of Cuban Association for Pattern Recognition.

Rafael Berlanga-Llavori is a professor in the Computer Science career at University Jaume I, Spain for 12 years. He received the B.S. degree from Universidad de Valencia in Physics, and the Ph.D. degree in Computer Science in 1996 from the same university. He is author of several articles in international journals, such as Information Processing & Management, Concurrency: Practice and Experience, Applied Intelligence, among others, and numerous communications in international conferences such as DEXA, ECIR, CIARP, etc.

José Ruiz-Shulcloper is a Senior Research at the Advanced Technologies Application Center, Cuba. He received the B.S. degree from Havana University in 1972 and the Ph.D. degree from the Lomonosov Moscow State University in 1978 both in Mathematics. He is author of several articles in international journals, such as Pattern Recognition, Pattern Recognition Letter, Fuzzy Sets & Systems, among others. He is founder and President of the executive board of Cuban Association for Pattern Recognition.