Available online at www.sciencedirect.com

SCIENCE DIRECT®

Applied Soft Computing 4 (2004) 149–158

Applied Soft Computing

www.elsevier.com/locate/asoc

Clustering terms in the Bayesian network retrieval model: a new approach with two term-layers

Luis M. de Campos*, Juan M. Fernández-Luna, Juan F. Huete

Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S.I. Informática,
Universidad de Granada, E-18071 Granada, Spain

Received 30 December 2002; received in revised form 19 September 2003; accepted 11 November 2003

Abstract

The retrieval performance of an information retrieval system usually increases when it uses the relationships among the terms contained in a given document collection. However, this creates the problem of how to obtain these relationships efficiently, and how to then use them to retrieve documents given a user's query. This paper presents a new retrieval model based on a Bayesian network that represents and exploits term relationships, overcoming these two drawbacks. An efficient learning method to capture these relationships, based on term clustering, as well as their use for retrieval purposes, is also shown.

Keywords: Bayesian networks; Information retrieval models; Learning; Term clustering

1. Motivation

The Information Retrieval process contains a certain amount of uncertainty in each one of its stages: a query is a vague description of the user needs, document representations are an incomplete characterisation of their contents and in the document retrieval process the uncertain component exists in the assessment of the degree of relevance of a document to a given query.

In the last decades, Bayesian networks [17] have become one of the most promising methodologies to manage uncertainty. Bayesian networks combine a qualitative representation of the problem, by means of a graphical representation of the dependences (and also independences) between the variables involved in the problem, with quantitative representation of the uncertainty, using a probabilistic approach. The main advantage of this formalism is that it can be performed efficiently by probabilistic computing.

Taking into account the two facts previously mentioned, the advantage of using Bayesian network methodology to solve Information Retrieval problems becomes clear [6,8,9,19,22]. Thus, the *Bayesian network retrieval model* (BNRM) [8] is a new model that aims to exploit the advantages of this kind of probabilistic graphical model in order to accurately represent knowledge, in this case that contained in a text collection, and its posterior efficient manipulation. The final objective is to use the network to retrieve documents in response to a user's query, computing the probability of relevance of each document in the collection given the query. Nevertheless, and

* Corresponding author. Tel.: +34-958-244019;
fax: +34-958-243-317.
E-mail addresses: lci@decsai.ugr.es (L.M. de Campos), jmfluna@decsai.ugr.es (J.M. Fernández-Luna), jhg@decsai.ugr.es (J.F. Huete).

1568-4946/$ – see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2003.11.003

mainly due to the large number of variables (terms and documents) in the process, the development of new methods to make the problem tractable and to obtain low time-solution costs without losing precision, becomes necessary.

The BNRM is composed of two different but related parts: The first stores all the documents from the collection (document subnetwork), and the second the terms contained in the documents (term subnetwork). This paper focuses mainly on the structure of the term subnetwork. In particular, our aim is to provide this structure with the capability of representing the term relationships in the collection. Thus, a more accurate representation of the collection is obtained and a better retrieval performance may be expected. However, considering term relationships implies additional effort owing to the fact that the best relationships have to be acquired and later used when a user formulates a query, causing an added delay in the retrieval time.

In terms of Bayesian networks, the previously mentioned acquisition stage implies the application of a learning algorithm. The topology to store these can be as complex as required, but taking into account the great number of terms that a common collection contains, the learning and posterior use (propagation) could be very time consuming.

In a previous study, when the model proposed [8] was designed, a polytree was chosen to represent term relationships. The main reason for this was that efficient learning and propagation algorithms exist for this kind of network topology. But using current collections, where the number of terms and documents is really very large, running a propagation algorithm, even for a polytree, and taking into account that in interactive Information Retrieval the user requires the system's answer in very few seconds, this kind of structure would not be the most appropriate.

Therefore, the main objective of this paper is to present a new and efficient method to determine the strongest relationships among terms in a collection, which could be classified as a term clustering technique. Its output will be an alternative topology for the term subnetwork also supporting the use of a very efficient propagation algorithm.

The remainder of the paper is structured as follows: Section 2 contains the basic knowledge about Information Retrieval and Bayesian networks needed to follow this paper. Next, Section 3 introduces the general topology of the retrieval model based on Bayesian networks. This is followed by Section 4, containing the new graph topology that will support the relationships among terms and how these relationships are captured. Sections 5 and 6 describe the assessment of the quantitative information that the Bayesian network must store and how the retrieval process is performed, respectively. In Section 7 the experimentation using several standard document collections and its results are presented. Finally, Section 8 contains the concluding remarks.

2. Preliminaries

Information retrieval (IR) is a subfield of computer science that deals with the automated storage and retrieval of documents [21]. An *IR system* is a computer program that matches user *queries* (formal statements of information needs) to documents stored in a database (the *document collection*). In our case, the *documents* will always be the textual representations of any data objects. An *IR model* is a specification about how to represent documents and queries, and how to compare them. Many IR models as well as the IR systems implementing them, such as the vector space model [21] or probabilistic model [3], do not use the documents themselves, but instead a kind of document surrogate, usually in the form of vectors of *terms* or *keywords*, which try to characterise the document's information content.[1] Queries are also represented in the same way.

When a user formulates a query, this is compared with each document from the collection and a score that represents its relevance (matching degree) is computed. Later, the documents are sorted in decreasing order of relevance and returned to the user.

To evaluate IR systems, in terms of retrieval effectiveness, several measures have been proposed. The most commonly used are *recall* (R) (the proportion of relevant documents retrieved), and *precision* (P) (the proportion of retrieved documents that are relevant for a given query). The relevance or irrelevance of a document is based, for test collections, on the *relevance judgements* expressed by experts for a fixed set of

[1] In the rest of the paper we will use the word *document* to denote both documents and document surrogates.

queries [21]. By computing the precision for a number of values of recall we obtain a recall-precision plot. If a single measure of performance is desired, the average precision for all the recall values considered may be used. Finally, if we are processing together a set of queries, the usual approach is to report mean values of the selected performance measure(s).

An interesting tool in IR is *clustering* [21], because it can be used to structure a document collection based on the similarities of the contained documents, thus supporting a user in a search for similar documents. This generic technique has also been applied to group terms, giving as possible results *thesauri*, structures that store the relationships among these terms. They can be useful to assist the user's formulation of a query, as well as to increase the effectiveness of the IR system, as is the case of this paper.

Probabilistic IR models use probability theory to deal with the intrinsic uncertainty with which IR is pervaded [9]. Also founded primarily on probabilistic methods, *Bayesian networks* [17] have been proved to be good models for uncertainty management, even in the IR environment, where they have already been successfully applied as an extension/modification of probabilistic IR models [7,19,22]. The networks are used to compute the posterior probabilities of relevance of the documents in the collection given a query.

Bayesian networks are graphical models capable of efficiently representing and manipulating n-dimensional probability distributions [17]. A Bayesian network uses two components to codify qualitative and quantitative knowledge:

- A *directed acyclic graph* (DAG), $G = (V, E)$, where the nodes in V represent the random variables from the problem we want to solve, and the topology of the graph (the arcs in E) encodes conditional (in)dependence relationships among the variables (by means of the presence or absence of direct connections between pairs of variables);
- A set of conditional probability distributions drawn from the graph structure: for each variable $X_i \in V$ we have a family of conditional probability distributions $P(X_i|\mathrm{pa}(X_i))$, where $\mathrm{pa}(X_i)$ represents any combination of the values of the variables in $\mathrm{Pa}(X_i)$, and $\mathrm{Pa}(X_i)$ is the parent set of X_i in G.

From these conditional distributions we can recover the joint distribution over V:

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i|\mathrm{pa}(X_i)) \qquad (1)$$

This decomposition of the joint distribution results in important savings on storage requirements. It also allows probabilistic inference (propagation) to be performed (efficiently, in many cases), i.e., computing the posterior probability for any variable given some evidence about the values of other variables in the graph [17].

3. The topology of the Bayesian network retrieval model

In this section, the basic topology of the Bayesian network retrieval model is described: the variables included and their relationships.

The set of variables V in this model is composed of two different sets of variables, $V = T \cup D$: the set $T = \{T_1, \ldots, T_M\}$ of the M terms in the glossary (index) from a given collection and the set $D = \{D_1, \ldots, D_N\}$ of the N documents that compose the collection.[2] Each term variable T_i takes its values in the set $\{\bar{t}_i, t_i\}$, meaning that the term T_i is not relevant or is relevant, respectively. Similarly, the domain of each document variable D_j is $\{\bar{d}_j, d_j\}$, meaning respectively that document D_j is not relevant or is relevant, with respect to a query.

Term variables are arranged in what is called *term subnetwork*, and document variables in the *document subnetwork*. To determine the topology of the basic Bayesian network, we have taken into account that there is a link joining each term node T_i in the term subnetwork and each document node D_j included in the document subnetwork, whenever T_i belongs to D_j. Also, there are not links joining any document nodes D_j and D_k (the document subnetwork is composed of isolated document nodes). Finally, any document D_j is conditionally independent of any other document D_k when we know for sure the (ir)relevance values for all the terms indexing D_j. This means that the links

[2] We will use the notation T_i (D_j, respectively) to refer to both the term (document, respectively) and its associated variable and node.

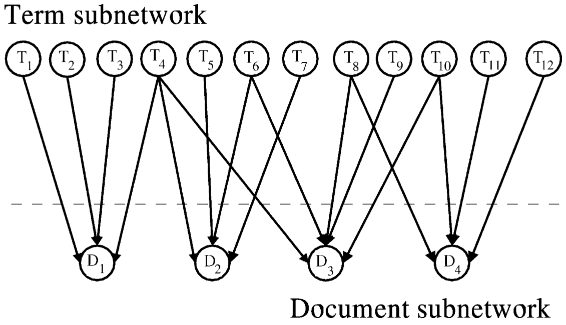

Fig. 1. The topology of the simple Bayesian network retrieval model.

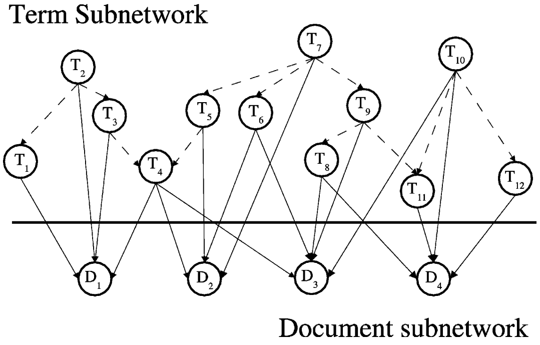

Fig. 2. The topology of the extended Bayesian network retrieval model.

joining term and document nodes have to be directed from terms to documents; therefore, the parent set of a document node D_j is the set of term nodes that belong to D_j, i.e., $\mathrm{Pa}(D_j) = \{T_i \in T | T_i \in D_j\}$.

The next step is to represent term relationships by means of a Bayesian network, i.e., specify the term subnetwork. If we do not consider term relationships, we will obtain the model proposed in [1] (an example is displayed in Fig. 1). On the other hand, a non trivial term subnetwork is considered in [8]; in this case, the subnetwork is a polytree (a graph in which there is no more than one undirected path connecting each pair of nodes) which is automatically constructed from the document collection. This topology is interesting because it supports exact and efficient inference algorithms that run in a time proportional to the number of nodes [17]. The learning algorithm, described in [6], is based on the algorithms proposed in [5,18], but includes several modifications and new contributions to adapt it to the IR environment. Fig. 2 shows an example of this model. The retrieval performance of the latter is usually better than that of the former, due to the fact that capturing term to term relationships within a collection gives a more accurate representation of the collection.

Nevertheless, considering the size of actual collections, the decision to use a polytree to represent term relationships would not be the most appropriate. This is due to the fact that, for very large networks, even the efficient inference method used in [8] (Pearl's exact propagation algorithm for polytrees [17]) is not fast enough, without forgetting the effort needed to acquire the structure from the document collection. This is the problem that led us to look for an alternative topology to represent these relationships with a lighter learning stage and, more importantly, the possibility of a very fast propagation mechanism.

4. Learning the term subnetwork: clustering terms to obtain the best relationships

In this section, we present a new topology for representing term relationships. First, we shall describe the new topology and then how to learn it. Later, in Section 6 we shall show how an extremely efficient propagation scheme can be used on this topology, in order to compute the posterior probabilities of each term node.

4.1. Topology: two term-layers

In the new topology we shall include explicit dependence relationships between T_j and each term in $R_p(T_j)$ (the set of those p terms most closely related to T_j, measured in a certain way). The new graph will use two layers of nodes to represent the term subnetwork: we duplicate each term node T_k in the original layer to obtain another term node T'_k, thus forming a new term layer, T'. The arcs connecting the two layers go from $T'_i \in R_p(T_j)$ to T_j. Therefore, in the new Bayesian network the set of variables is $V = T \cup T' \cup D$. The parent set of any original term node $T_j \in T$ is defined as $\mathrm{Pa}(T_j) = R_p(T_j)$. We use this topology, a bipartite graph, because it will support a very fast propagation algorithm in the term subnetwork. The complete Bayesian network contains three simple layers (see

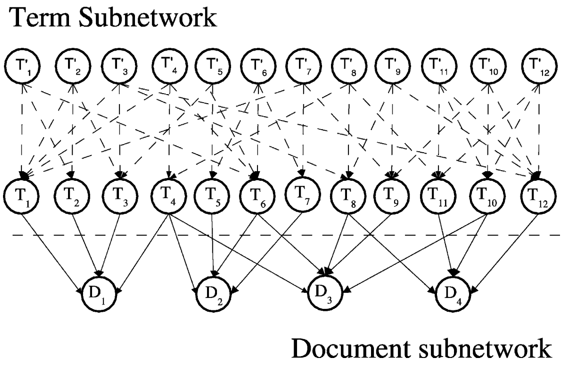

Fig. 3. Topology of the BNRM with two term-layers.

Fig. 3), without connections between the nodes in the same layer, and this fact will be essential for the efficiency of the inference process in the whole network.

4.2. Learning term relationships: clustering terms

To build the term subnetwork described previously, we have to determine which is the set $R_p(T_j)$, for each term node T_j, i.e., the set of terms that are most similar to T_j. Basically, this is a clustering process, in which we obtain groups of very similar terms. We have designed a relatively simple method to obtain $R_p(T_j)$ that we shall explain in the following paragraphs.

A useful measure to determine dependences between variables frequently used for learning Bayesian network structures, is the Kullback–Leibler's cross entropy [13]. Basically, the pairs of terms with the highest cross entropy should be connected in the graph. In fact, this measure is used within the polytree learning algorithm considered in the original BNR model [8]. However, the situation has altered, given that dependence is distinct from similarity, although they are related. A high value of cross entropy between two terms means high dependence, although this dependence may be positive (in the sense that these two terms mainly co-occur in the same documents) but also negative (when they do not occur in any common document). In our case, a good learning algorithm would need to include in $\mathrm{Pa}(T_j)$ only terms that are positively correlated with T_j hence we cannot use cross entropy to measure similarity between terms. Therefore, the approach presented in this paper is based directly on the frequency of co-occurrences of two terms. Before explaining further, let us introduce some notations:

Table 1
Contingency table for terms T_i and T_j

	$T_j = \bar{t}_j$	$T_j = t_j$	
$T_i = \bar{t}_i$	$n_{\bar{t}_i\bar{t}_j}$	$n_{\bar{t}_i t_j}$	$n_{\bar{t}_i}$
$T_i = t_i$	$n_{t_i\bar{t}_j}$	$n_{t_i t_j}$	n_{t_i}
	$n_{\bar{t}_j}$	n_{t_j}	N

Given a term T_j, if we want to know which terms, from the rest of the collection, are most closely related to it, for each one of these terms, T_i, the values displayed in Table 1 may be computed by counting frequencies.

In this case, t_i means "T_i occurs", and $\bar{t}_i$ stands for "T_i does not occur" (and respectively for T_j); $n_{\bar{t}_i\bar{t}_j}$ is the number of times in which neither T_i nor T_j occur in a document; $n_{t_i t_j}$ is the number of times in which both terms occur in the same document and, finally, $n_{\bar{t}_i t_j}$ and $n_{t_i\bar{t}_j}$, the number of documents in which only one of the two terms occurs.

The following expression, a maximum likelihood estimator, could be used to measure the strength of their co-occurrence relationship, from the perspective of the term T_j:

$$\mathrm{strength}(T_j, T_i) = \frac{n_{t_i t_j}}{n_{t_i}} \tag{2}$$

i.e. a coefficient that measures the ratio between the number of documents in which T_j co-occurs with T_i, with respect to the total number of documents in which T_i occurs. When this quotient is close to 1.0, this means that almost all the documents indexed by T_i are also indexed by T_j so that T_j is quite similar to T_i. But anomalous behaviour is observed in some cases: if, for instance, T_i occurs only in one document and T_j occurs in that document, the result would be 1.0; on the other hand, if we have the case in which T_i and T_j share five documents, of the five in which T_i is in the collection, the ratio is the same, but we would say that T_i and T_j are more closely related in the second example, although the value obtained by Eq. (2) is the same in both cases. To solve this problem, we will use a Bayesian estimator [2] instead of the maximum likelihood estimator:

$$\mathrm{strength}(T_j, T_i) = \frac{n_{t_i t_j} + 1}{n_{t_i} + 2} \tag{3}$$

When this estimator is used, a new problem arises: imagine a pair of terms such as $n_{t_i t_j} = 0$, i.e., they do not co-occur in any document, and also $n_{t_i} = 1$, then strength(T_j, T_i) would be 0.33, a value that would be always greater than the one obtained when these terms co-occur once, for instance, and $n_{t_i} > 4$, a situation that is not very logical. To solve this problem, we have employed a modified strength function, strength$'(T_j, T_i)$, that will be 0 when $n_{t_j t_i} = 0$, and strength(T_j, T_i), otherwise.

$$\text{strength}'(T_j, T_i) = \begin{cases} 0, & \text{if } n_{t_i t_j} = 0 \\ \text{strength}(T_j, T_i), & \text{otherwise} \end{cases} \tag{4}$$

Therefore, to learn the term subnetwork implies to discovering which terms have the strongest relationships with each one of the terms in the collection, i.e., to determine the sets of parents $R_p(T_j)$, $T_j \in T$. Thus, for each T_j, the measure strength(T_j, T_i), $\forall T_i \in T$ is computed. The duplicates of the p terms with the highest values are selected to be elements of $R_p(T_j)$. We have to highlight that the equivalent term to T_j in T', T'_j, is always included in $R_p(T_j)$, i.e., a term is always related to itself.

4.3. Related works based on soft computing

In this section, a brief overview of some Soft Computing techniques applied to clustering is presented.[3] Although there is a great amount of work on document clustering, we shall focus our attention on term clustering and thesaurus construction given the theme of this paper.

Starting with Bayesian networks, Jing and Croft [10] have developed an association thesaurus using INQUERY [22] as an underlying search engine to access this structure. It considers a basic method of counting co-occurrences between phrases and terms as a measure of association. A second application of Bayesian networks to this area is [15]. In this case, using a similar expression to Eq. (2) the authors build a sigmoid Bayesian network in the form of a collocation map. A final example is [6], where a thesaurus is built based on Kullback–Leibler's cross entropy in the form of a Bayesian network. These three applications use the thesaurus to carry out query expansion tasks, measuring the effectiveness of this technique when terms are extracted from thesauri and added to the original query. This use is one of the main differences with respect to our approach, given that in the BNR Model the relationships are included directly in the Bayesian network and used implicitly in retrieval time, as will be explained later.

Continuing with other soft computing techniques, one of the most closely related applications are the self-organising maps (SOM) [12], a special type of neural network that clusters document or term vectors according to a similarity measure. The clusters are based on neighbourhood relations, in such a way that terms of nearby clusters are usually more similar than others in more distant clusters. These groups are mapped on two dimensions so a very intuitive visual representation is given. WEBSOM [11,14] is one of the main applications based on SOM. In this case, it is used to organise very large collections of Web documents and is composed of two maps: the first learns the relations of the terms in the text, and the second is used to classify documents according to the relationships of the terms they contain. Also, thesauri have been developed using such maps. An example of application of SOM to a thesaurus generation is the algorithm Scalable SOM (SSOM) [20], by which the semantic relationships between terms are extracted and a hierarchy of categories generated.

5. Specifying qualitative information in the BNR Model

Once the graph has been built, the probability distributions stored in each node of the network must be estimated. Thus, all the root nodes, i.e., those which do not have parents, will store marginal distributions. In our specific case, the only nodes of this type are term nodes placed in the first term layer. For each root term node, we have to assess $p(t_i)$ and $p(\bar{t}_i)$; we use the following estimator: $p(t_i) = (1/M)$ and $p(\bar{t}_i) = 1 - p(t_i)$ (M is the number of terms in the collection).

The nodes with parents (term and document nodes) Will store conditional probability distributions, one for each of the possible configurations that their parent nodes can take. Terms nodes in the second term layer

[3] A more general review of the application of soft computing techniques to information retrieval can be found in [4,16].

must store the conditional probabilities $p(T_i|\mathrm{pa}(T_i))$, where $\mathrm{pa}(T_i)$ is a configuration of values associated to the set of parents of T_i. Analogously, document nodes must store $p(D_j|\mathrm{pa}(D_j))$.

Starting from document nodes, the estimation of the conditional probabilities of relevance of a document D_j, $p(d_j|\mathrm{pa}(D_j))$, is not an easy problem. The reason is that the number of conditional probabilities that we need to estimate and store for each D_j grows exponentially with the number of parents of D_j. Instead of explicitly computing and storing these probabilities, we use a *probability function* (also called a canonical model of multicausal interaction [17]). Each time that a given conditional probability is required during the inference process, the probability function will compute and return the appropriate value. We have developed a new general canonical model: for any configuration $\mathrm{pa}(D_j)$ of $\mathrm{Pa}(D_j)$ (i.e., any assignment of values to all the term variables in D_j), we define the conditional probability of relevance of D_j as follows:

$$p(d_j|\mathrm{pa}(D_j)) = \sum_{T_i \in D_j, t_i \in \mathrm{pa}(D_j)} w_{ij} \qquad (5)$$

where the weights w_{ij} verify that $0 \leq w_{ij}$, $\forall i, j$ and $\sum_{T_i \in D_j} w_{ij} \leq 1 \forall j$. The expression $t_i \in \mathrm{pa}(D_j)$ in Eq. (5) means that we only include in the sum those weights w_{ij} such as the value assigned to the corresponding term T_i in the configuration $\mathrm{pa}(D_j)$ is t_i. So, the more terms that are relevant in $\mathrm{pa}(D_j)$ the greater the probability of relevance of D_j. The specific weights, w_{ij} used in this paper by our models, for each document $D_j \in D$ and each term $T_i \in D_j$, are:

$$w_{ij} = \alpha^{-1} \frac{\mathrm{tf}_{ij}\,\mathrm{idf}_i^2}{\sqrt{\sum_{T_k \in D_j} \mathrm{tf}_{kj}\,\mathrm{idf}_k^2}} \qquad (6)$$

where α is a normalising constant (to assure that $\sum_{T_i \in D_j} w_{ij} \leq 1 \forall D_j \in D$. tf_{ij} is the *term frequency* of the term T_i in document D_j and idf_i is the *inverse document frequency* of the term T_i in the whole collection. Obviously, many other weighing schemes are possible. The weights in Eq. (6) have been chosen to resemble the well-known cosine measure [21].

Finally, we have to define the conditional probabilities $p(T_j|\mathrm{pa}(T_j))$ for the terms in the original term layer T. For the same reasons we used probability functions in the document layer, we use a probability function belonging to the general canonical model defined in Eq. (5), where the weights v_{ij} measure the influence of each $T'_j \in \mathrm{Pa}(T_j)$ on term T_j:

$$p(t_j|\mathrm{pa}(T_j)) = \sum_{T'_i \in \mathrm{Pa}(T_j), t'_i \in \mathrm{pa}(T_j)} v_{ij} \qquad (7)$$

Our proposal for the weights v_{ij} in Eq. (7) is the following:

$$v_{ij} = \frac{1-\beta}{s_j} \mathrm{strength}'(T_j, T_i), \quad \forall T'_i \in \mathrm{Pa}(T_j), \quad i \neq j \qquad (8)$$

$$v_{ij} = \beta$$

where $S_j = \sum_{T'_i \in \mathrm{Pa}(T_j), i \neq j} \mathrm{strength}'(T_j, T_i)$ and β is a parameter, $0 < \beta < 1$, that is used to control the importance of the contribution of the term relationships being considered for a term T_j to its final degree of relevance. In this way we are imposing a uniform upper boundary for the importance of this combination equal to $1 - \beta$.

6. Retrieving documents: inference in the BNR Model

Given a query Q submitted to our system, the retrieval process starts placing the evidence in the term subnetwork: the state of each term T'_{iQ} belonging to Q is fixed to t'_{iQ} (relevant). Then the inference process is run in the whole network obtaining, for each document D_j, its probability of relevance given that the terms in the query are also relevant, $p(d_j|Q)$. Finally, the documents are sorted in decreasing order of probability to carry out the evaluation process.

Taking into account the large number of nodes in the Bayesian network and the fact that it contains cycles and nodes with a great number of parents, general purpose inference algorithms cannot be applied due to efficiency considerations, even for small document collections. To solve this problem, we have designed a specific inference method that takes advantage of both the topology of the network and the kind of probability functions used for document and terms nodes (Eqs. (5) and (7)). This method is composed of two stages, and constitutes an exact propagation algorithm (by virtue of the properties of the canonical model being used and the layered topology of the network [8]):

(1) The computation of the posterior probability of relevance of the term nodes belonging to T, $p(t_j|Q), \forall T_j$, which is carried out by simply evaluating the following expression:

$$p(t_j|Q) = \sum_{T_i' \in \mathrm{Pa}(T_i)} v_{ij}\, p(t_i'|Q) \tag{9}$$

Notice that $p(t_i'|Q) = 1.0$ if $t_i' \in Q$, and $1/M$ otherwise (because terms in the T' layer are marginally independent, the posterior probability of the terms which are not in the query coincides with their prior probability, $p(t_i'|Q) = p(t_i') = 1/M$ and the probability of the query terms is equal to 1); by substituting the weights v_{ij} in Eq. (8), the final expression for the calculation of $p(t_j|Q)$ is:

$$p(t_j|Q) = \frac{1-\beta}{S_j} \sum_{T_i' \in \mathrm{Pa}(T_j), i \neq j} \mathrm{strength}'(T_j, T_i)\, p(t_i'|Q) + \beta p(t_j'|Q) \tag{10}$$

(2) The evaluation of the posterior probability of relevance of the document nodes, $p(d_j|Q)$ which can be carried out using the information obtained in the previous step, in the following way:

$$p(d_j|Q) = \sum_{T_i \in \mathrm{Pa}(D_j)} w_{ij}\, p(t_i|Q) \tag{11}$$

Therefore, the propagation with this topology is reduced to evaluate Eqs. (10) and (11), giving as a result a very efficient retrieval method.

7. Experiments and results

To test the new Bayesian network topology, we have run several retrieval experiments with three medium-size standard collections: ADI, CISI, and CRANFIELD. The main characteristics of these collections with respect to number of documents, terms and queries are (in this order): ADI (82, 828, 35), CISI (1460, 4985, 76), and CRANFIELD (1398, 3857, 225).

The baseline for comparing the results of the experiments performed with the new two term-layers network is the original BNR model, where the topology of the term subnetwork is a polytree [8]. Therefore, our aim is to compare the effectiveness of the two topologies, the original and the new. In order to carry out this task, we have performed tests with a different number of parents, p, for the terms in T, and for several values of the parameter β. To be exact, the first parameter has been set to 5, 10, and 15 parents; the second to 0.6, 0.7 and 0.8. The performance measure considered is the average precision for the 11 standard values of recall.

Table 2
Results of the experiments with the new topology of the term subnetwork

p	β	ADI	CISI	CRANFIELD	AP-11
		0.4130	0.2007	0.4314	(BNRM)
5	0.6	0.4524	0.216	0.4314	AV-11p
		9.54	7.62	0.00	%C
5	0.7	0.4547	0.2212	0.4332	AV-11p
		10.10	10.21	0.42	%C
5	0.8	0.4676	0.2207	0.4316	AV-11p
		13.22	9.97	0.05	%C
10	0.6	0.4587	0.2182	0.4334	AV-11p
		11.07	8.72	0.46	%C
10	0.7	0.4681	0.22	0.4347	AV-11p
		13.34	9.62	0.76	%C
10	0.8	0.4695	0.221	0.4331	AV-11p
		13.68	10.11	0.39	%C
15	0.6	0.4678	0.2211	0.4332	AV-11p
		13.27	10.16	0.42	%C
15	0.7	0.4651	0.2203	0.434	AV-11p
		12.62	9.77	0.60	%C
15	0.8	0.468	0.2208	0.4329	AV-11p
		13.32	10.01	0.35	%C

The results of this experimentation are presented in Table 2, where the average precision values for the 11 standard recall points of the original BNR model, for each collection, are shown in the second row (Noted as ‘*AP-11p* (*BNRM*)’).

The average precision (*AV-11p*) of the experiments run with the new model for different values of the number of parents and the parameter β (Labels p and β, respectively, in the table) are also shown, as well as the percentage of change with respect to the corresponding average precision in the original model (%*C*).

Although the results are sensitive to the values of the two parameters, p and β, they do not vary greatly.

In fact, the means and standard deviations of the percentage of change are, respectively, 12.24 and 1.49 for ADI, 9.58 and 0.81 for CISI, and 0.38 and 0.23 for CRANFIELD. We believe that the number of parents, *p*, should not be low (because this could prevent the inclusion of useful term relationships). Analogously, with respect to β, this parameter should not be low (since in this case, the term relationships on a given term could be overloaded).

We can also observe that the effectiveness of the new model is even better than the performance of the polytree-based model in terms of retrieval success, at least for the three collections considered. This is a positive side effect, because our initial goal was to increase the efficiency without degrading the effectiveness.

8. Concluding remarks

In this paper a new topology for representing term relationships, based on a term clustering method, has been presented. Instead of using a polytree as the underlying structure of the term subnetwork, we have designed a new graph, a bipartite graph (two layers of nodes representing the terms in the collection), that stores the strongest relationships among terms. The main advantage of this graph is that the exact propagation that had to be carried out in the original polytree is replaced by the evaluation of simple expressions, resulting in a very efficient method. The main application of this new model will be the retrieval of TREC documents, where, taking into account its topology and the whole inference method, we think that it will be competitive and efficient.

We have shown with the experiments that this new model in which two term-layers are used to encode term relationships behaves better than the original model, although this depends on the collection being tested.

There are several ways in which the model may be modified to improve its performance. The first is the design of more accurate ways of determining the strength of the relationships among terms, reflecting only positive dependences and, at the same time, using this previous measure or designing a new one, to develop a method to select the best terms. This selection could be based only on co-occurrences or a combined way between co-occurrences and the cross entropy. To be completely sure that the terms are dependent, we could carry out an independence test. A second aspect related to this point is the decision regarding the number of parents of each term. It would be more reasonable that this number were not the same for all the terms, being a term-dependent parameter. Also, the design of a new and more sophisticated probability function to be evaluated in the original layer of terms should be completed, in which the β parameter is removed. An alternative way is the use of SOM to mine the relationships among terms. With these tools, once the map of terms has been generated, the two-layer network could be created linking terms which are adjacent in the map.

Acknowledgements

This work has been jointly supported by the Spanish MCYT, FIS and Consejería de Salud de la Junta de Andalucía, under Projects TIC2000-1351, PI021147 and 177/02, respectively.

References

[1] S. Acid, L.M. de Campos, J.M. Fernández, J.F. Huete, An information retrieval model based on simple Bayesian networks, Int. J. Intell. Syst. 18 (2003) 251–265.

[2] B. Cestnik, Estimating probabilities: a crucial task in machine learning, in: Proceedings of ECAI Conference, 1990, pp. 147–149.

[3] F. Crestani, M. Lalmas, C.J. van Rijsbergen, L. Campbell, Is this document relevant? ... probably. A survey of probabilistic models in information retrieval, ACM Comput. Surv. 30 (4) (1991) 528–552.

[4] F. Crestani, G. Pasi, Soft Information retrieval: applications of fuzzy set theory and neural networks, in: N. Kasabov, R. Kozma (Eds.), Neuro-Fuzzy Techniques for Intelligent Information Systems, Physica Verlag, 1999, pp. 287–315.

[5] L.M. de Campos, Independency relationships and learning algorithms for singly connected networks, J. Exp. Theor. Artif. Intell. 10 (4) (1998) 511–549.

[6] L.M. de Campos, J.M. Fernández, J.F. Huete, Query expansion in information retrieval systems using a Bayesian network-based thesaurus, in: Proceedings of the 14th UAI Conference, 1998, pp. 53–60.

[7] L.M. de Campos, J.M. Fernández, J.F. Huete, Building Bayesian network-based information retrieval systems, in: 2nd LUMIS Workshop, 2000, pp. 543–552.

[8] L.M. de Campos, J.M. Fernández, J.F. Huete, The BNR model: foundations and performance of a Bayesian network-

based retrieval model, Int. J. Approx. Reasoning 34 (2003) 265–285.

[9] R. Fung, B. Del Favero, Applying Bayesian networks to information retrieval, Commun. ACM 38 (2) (1995) 42–57.

[10] Y. Jing, W. Bruce Croft, An association thesaurus for information retrieval, in: Proceedings of RIAO-94, 4th International Conference, 1994, pp. 146–160.

[11] S. Kaski, T. Honkela, K. Lagus, T. Kohonen, WEBSOM—self-organizing maps of document collections, Neurocomputing 21 (1998) 101–117.

[12] T. Kohonen, Self-organized formation of topologically correct feature maps, Biol. Cybernetics 43 (1982) 59–69.

[13] S. Kullback, R.A. Leibler, On information and sufficiency, Ann. Math. Stat. 22 (1951) 76–86.

[14] K. Lagus, T. Honkela, S. Kaski, T. Kohonen, WEBSOM for textual data mining, Artif. Intell. Rev. 13 (5–6) (1999) 345–364.

[15] Y.C. Park, K. Choi, Automatic thesaurus construction using Bayesian networks, Information Process. Manage. 32 (5) (1996) 543–553.

[16] S. Pal, V. Talwar, P. Mitra, Web mining in soft computing framework: relevance, state of the art and future directions, IEEE Trans. Neural Netw. 13 (5) (2002) 1163–1177.

[17] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan and Kaufmann, 1988.

[18] G. Rebane, J. Pearl, The recovery of causal polytrees from statistical data, Uncertainty in Artificial Intelligence, 1989, pp. 175–182.

[19] B.A. Ribeiro-Neto, R.R. Muntz, A belief network model for IR, in: H. Frei, D. Harman, P. Schäble, R. Wilkinson (Eds.), 19th ACM-SIGIR Conference, ACM, 1996, pp. 253–260.

[20] D. Roussinov, H. Chen, A scalable self-organizing map algorithm for textual classification: a neural network approach to thesaurus generation, Commun. Cogn. 15 (1–2) (1998) 81–112.

[21] G. Salton, M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, 1983.

[22] H.R. Turtle, W.B. Croft, Evaluation of an inference network-based retrieval model, Information Syst. 9 (3) (1991) 187–222.