Building Bayesian Network-Based Information Retrieval Systems

Luis M. de Campos
Departamento de Ciencias
de la Computación e I.A.
E.T.S.I. Informática
Universidad de Granada
18071 - Granada, Spain
lci@decsai.ugr.es

Juan M. Fernández
Departamento. de Informática.
Escuela Politécnica Superior
Universidad de Jaén
23071 - Jaén, Spain
jmfluna@ujaen.es

Juan F. Huete
Departamento. Ciencias
de la Computación e I.A.
E.T.S.I. Informática
Universidad de Granada
18071 - Granada, Spain
jhg@decsai.ugr.es

Abstract

Bayesian networks are suitable models to deal with the Information Retrieval problem because they are appropriate tools to manage the intrinsic uncertainty with which this area is pervaded. In this paper we introduce several modifications to the previous works on this field, adding new features and showing how a good retrieval effectiveness can be achieved by improving the quality of the Bayesian networks used in the model and tuning some of their parameters.

Keywords: *Information retrieval, Bayesian networks, Learning, Conditional probability functions*

1 Introduction

In our days in which Internet and the World Wide Web is getting more and more relevance in our life, the *Information Retrieval* (IR) field is strongly centering the attention of research, changing the natural environment in which IR had been moving since the 40s, when it was born. But despite this change, and the influence of new technologies, the foundation of IR is still the same: people need information to solve problems and this field tries to organize all the required things to provide that information in the best conditions.

Therefore, IR deals with the following topics [17]: Representing the information to be easily handled, storing and organizing that representation, facilitating the posterior accessing, and retrieval of the information required by an user's information need.

Also, the stages which compose this previous cycle of retrieval will be carried out by means of a software in a computer, which will be called *Information Retrieval System* (IRS). In our case, the IRS will deal with texts in any of its varieties, i.e., e-mails, books, research articles, etc. generically referred to as *documents*.

In this paper we introduce an IRS based on a *Bayesian network model* [1]. The retrieval engine of our model is composed of a Bayesian network with two layers of nodes, representing the documents and the terms in the collection. In a first approach, we assume the independence between terms. This assumption is unrealistic, and will be relaxed in the second approach, where the relationships between the terms are allowed playing the role of a thesaurus. We show how, by improving the quality of the term layer adding the thesaurus, the performance of the IRS is also improved.

After specifying the structure of the Bayesian network there still remains the problem of assessing the probability distributions stored in the nodes. For document nodes, it is necessary to estimate a very big conditional probability table, which arises problems related to efficiency and storage capacity. Also, the management of these tables by the propagation modules becomes a time-consuming process. We propose in this work an alternative way to estimate the probability matrices allocated in the nodes, lightening the previously mentioned problems.

For term nodes, whose conditional probability matrices are estimated from the whole collection, a technique based on relaxing the conditional probabilities is presented. This method demonstrates how by finding the correct parameters, the IRS can obtain better results. Also, the way in which the prior probabilities are computed is a relevant fact that influences the retrieval effectiveness. To expose these ideas, this paper is divided into six sections. The next section deals with all the concepts related to IR that are necessary to understand the rest of the paper, and also introduces the Bayesian network models that have been already developed. Section 3 shows the general structure of our Bayesian networks. In section 4, the specific details about probability

[1] An *IR model* is a specification about how to represent documents and queries, and how to compare them.

0-7695-0680-1/00 $10.00 © 2000 IEEE

distributions stored in the networks are discussed. In section 5, we expose how we have carried out the experimentation and the results we have obtained with our model. Finally, the last section contains the concluding remarks and some proposal for future research.

2 Preliminaries: Information Retrieval and Bayesian network models

Given a document collection, the first step to operate with an IRS, as previously mentioned, is to characterize the content of the document, task called *indexing*. It obtains a representation of each document in a suitable form to be managed by a computer. The result is a set of keywords or terms extracted from each text that should appropriately express the content of the document: $D_j = (t_{1j}, t_{2j}, ..., t_{kj})$, although this representation depends on the IR model being used. Because they are not equally important, these terms could be weighted to highlight their importance in the documents they belong, as well as in the whole collection. A weighted indexed document could be $D_j = ((t_{1j}, w_{1j}), (t_{2j}, w_{2j}), ..., (t_{kj}, w_{kj}))$, where each w_{rj} is the weight associated to the corresponding term. Usually, we use the weight known as *tf.idf* weight. In this case, the value associated to a term is computed multiplying the frequency of the term in that document (*tf*) by the *inverse document frequency (idf)* of the term in the collection, $idf_i = \lg \frac{N}{n_i}$, where N is the number of documents in the collection, and n_i is the number of documents that contain the i^{th} term. The $tf.idf$ weight increases proportionally to the frequency of the term in the document and decreases proportionally to the number of documents containing the term.

When the indexing process has finished and the collection is ready to be used, a user interacts with the IRS by means of a query. That query is a description of the user's information need, and must be also indexed to produce a representation which can be handled by the system. The next step is the retrieval of those documents which are the most relevant to the query. The matching process is based on the search strategy implemented by the corresponding model. The result of this stage is a ranking of documents sorted by the proximity of each document to the query. There are three classic retrieval models: Boolean, vector space and probabilistic models [16, 17].

In this paper we focus on probabilistic models [7], which estimate the probability that a document satisfies a query, and mainly differ in the way in which they compute that probability of relevance. Based on probabilistic methods, Bayesian networks [14] have been proved to be a good model to manage uncertainty, even in the IR environment, in which they have been successfully applied as an extension of the probabilistic IR model, because they offer important advantages to deal with the intrinsic uncertainty with which IR is pervaded [9, 5].

A Bayesian network is a *Directed Acyclic Graph* (DAG), where the nodes represent the variables from the problem we want to solve. In that kind of graph, the knowledge is represented in two ways [14]: *(a)* Qualitatively, showing the (in)dependencies between the variables, and *(b)* Quantitatively, expressing the strength with which we believe the dependence relationships, and represented using a set of conditional probability distributions.

We are briefly going to review the different works developed based on Bayesian networks. Croft and Turtle have carried out an important work in this field [5, 18]. Their model is based on a Bayesian network formed by two pieces: the collection network and the query network. The former is composed of two types of nodes which represent documents and terms (concepts). The latter is built to represent the queries submitted to the system by means of query nodes and query concept nodes. The document network is fixed for a given document collection, and the query network is created each time that a user formulates a query. Once the probabilities have been assessed for each node, inference is carried out instantiating each document node, in turn. Therefore, the probability that the query is met given that a document has been observed in the collection, is obtained. After all the propagation processes, the posterior probabilities are sorted in decreasing order, so the higher the probability the more relevant the document is.

Closely related to this work, Ghazfan et al. in [10, 12] explain their modifications to Croft and Turtle's network, with the only goal of giving a '*corrected semantic meaning for the inference process*'. Thereby, they change the direction of all the arcs, instantiating only the query node instead of each document and propagating only once towards document nodes.Ribeiro and Muntz [15] have also developed a Belief network model, which generalizes the classical models.

3 Description of our IR models based on Bayesian networks

Our model is composed of a DAG, where two different sets of nodes can be found: a set containing binary random variables representing the terms in the glossary from a given collection, and a second, corresponding also to binary random variables, but in this case related to the documents which belong to the collection. The document set will be referred to as the *document subnetwork*, and, similarly, the term set will be called *term subnetwork*.

On the one hand, a variable T_i associated to a term takes its values from the set $\{\bar{t}_i, t_i\}$, where $\bar{t}_i$ stands for 'the term T_i is not relevant', and t_i represents 'the term T_i is relevant'. On the other hand, a variable referring to a document D_j,

has its domain in the set $\{\bar{d}_j, d_j\}$, where in this case, $\bar{d}_j$ and d_j respectively mean 'the document D_j is not relevant for a given query', and 'the document D_j is relevant for a given query'.

The nodes from both subnetworks are connected between them by means of links, in such a way that for each term that has been used to index a document, there is a link between the node representing the keyword and the node associated with the document to which the term belongs.

Because we are working with DAGs, there is a question that immediately arises: What is the orientation of these links to form arcs of a DAG? There are two possible answers: *(a)* from documents to term nodes, supported by Croft and Turtle [5, 18], *(b)* the arcs will leave the term nodes to point to the document nodes, which is maintained by Ghazfan et al. [10, 12] and Ribeiro and Muntz [15].

In our case, we think that it is more intuitive to consider the probability of relevance of a document given a query, which leads to adopt the second option and therefore, the arcs joining both subnetworks in our model will follow the direction described in *(b)*.

Until now, we have not discussed anything about the relationships between the term nodes in the term subnetwork, and similarly, the relationships inside the document layer. First of all, let briefly concentrate on the latter.

We will assume that the relation between documents only occurs through the terms included in these documents. Therefore, the document nodes will only receive the arcs from term nodes and not from other document nodes. This assumption implies that documents are conditionally independent given that we know the terms that they contain. Nevertheless, if we could have some semantic information relating two documents then the relationships between documents would be also allowed.

With regard to the relationships between term nodes, we have considered two hypotheses: *(a)* the terms are independent between each other, and *(b)* the terms are dependent. In the first case, our model is going to be composed of two simple node layers, the term and document subnetworks, arcs only exist from nodes in the first subnetwork to nodes in the second one. The second case will imply a Bayesian network learning process that captures the relationships among terms. In the two following subsections we will describe both models, specifying the main characteristics that define them.

3.1 The simple retrieval Bayesian network

This model is composed of two subnetworks inside of which the nodes are independent, existing links between the term nodes and the document nodes, which join the nodes from both subnetworks. All the terms nodes are root nodes, having prior probabilities stored in them, and all the document nodes are leaf nodes and should contain a conditional probability matrix. A graphical representation of the simple retrieval Bayesian network is shown in figure 1.

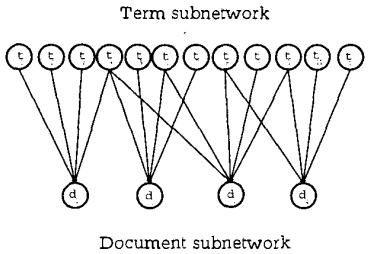

Figure 1. The simple IR Bayesian network.

The structure of our simple IR Bayesian network is similar to the network introduced by Ghazfan et al. [10, 12] and Ribeiro and Muntz [15]. As we will see later, the differences are mainly in the probability distributions stored.

3.2 The augmented retrieval Bayesian network

The major change with respect to the simple model is that the term subnetwork is not composed of a set of independent term nodes, but a more complex graph, showing the most important (in)dependences between the terms in the collection, resulting as a consequence in a more accurate model. To be precise, the graph that has been configured as the base of the term subnetwork is a polytree – a DAG where there is no more than one undirected path connecting each pair of nodes.

This new model needs a learning phase which uses an inverted file for the collection. Then, by means of a learning algorithm, a Bayesian network is constructed. Basically, what we are automatically building is a kind of thesaurus[2] for the collection. This thesaurus was used by the authors in a previous work [2] to develop a query expansion system, in which the most relevant terms to those belonging to the original query, selected after propagating the evidences in the polytree, were added to the query. The result of that stage was passed to the Smart Retrieval System [4], to finally carry out the retrieval process, getting the relevant documents for that new expanded query.

In the IR field the number of terms in a collection is very big. Therefore, the main reason why we chose a polytree was because the bigger and more complex the learned graph is, the more time is consumed for learning it and carrying out the propagation process. For singly connected graphs, like polytrees, there is a set of very efficient learning and propagation algorithms that reduce both times, without losing too much accuracy and expressiveness.

The learning algorithm is based on Chow and Liu's

[2] A structure which stores the terms and the relationships between them, used to improve the IRS performance.

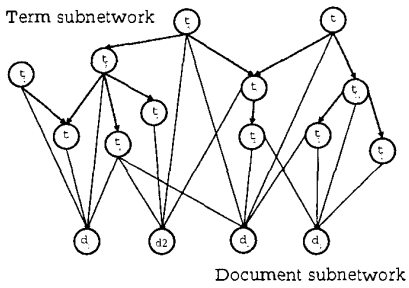

Figure 2. The augmented IR Bayesian network.

method for constructing dependence trees ([3], adding some additional features.

Figure 2 represents an example of the augmented Bayesian network model, where we notice that the connection between terms and document nodes is exactly the same as described previously.

4 Probability matrices in the Bayesian network models

Once we know the structure of the two Bayesian network models, we are going to focus our attention on the probability matrices stored in them.

There are two kinds of nodes in our Bayesian network: term and document nodes. For the first type, the prior distribution must be specified for all the nodes which are root nodes. Therefore, given a root node representing a term T_i, it must store the probability that T_i is not relevant and the probability that T_i is relevant. Both types of nodes which own parents will store conditional probability matrices.

When the probability matrices are going to be computed, the next three points must be taken into account: (1) The way of calculating the prior probabilities, (2) the size of the conditional probability matrices, and (3) the computation of the conditional probabilities.

In the following subsections we will deal with these three important questions, which are completely relevant for the retrieval effectiveness, and must be correctly tuned to achieve good results.

4.1 Computation of the prior probabilities

We have considered three different ways of computing prior probabilities:

- $pp1$: Given a term T_i, the probabilities which will store its related node are $p(t_i) = 0.5$ and $p(\bar{t}_i) = 0.5$, i.e., the uniform distribution.
- $pp2$: $p(t_i) = \frac{1}{M}$ and $p(\bar{t}_i) = 1 - p(t_i)$, M being the number of terms in a given collection.
- $pp3$: $p(t_i) = \frac{n_i}{N}$ and $p(\bar{t}_i) = 1 - p(t_i)$, where N is the number of documents in the data set, and n_i is the number of documents in which the term T_i occurs.

What we were looking for designing these prior probability expressions is to observe whether the retrieval results were relevant to prior probabilities. $pp1$ and $pp2$ expressions compute the same values for all the terms, but $pp2$ gives a prior probability of relevance to any term much lower than $pp1$. On the other hand, $pp3$ assigns a different probability to each term depending on the number of occurrences of the term in the collection.

4.2 The size of the conditional probability matrices

When the number of parents of a given node is very big, conditional probability matrix computing turns to be a problem for several reasons: the time needed to obtain these matrices is very long, the matrix sizes are very big, with the subsequent problems with its storage, and finally, the propagation process gets slower because of the management of these matrices.

At this point in the paper, we are going to focus our attention on the conditional probability matrices placed in every document node. As previously mentioned, each document node will have as many parents as terms were assigned to it in the indexing process. Therefore, it should store 2^k conditional probability matrices, k being the number of parents. Taking into account that the average number of terms per document in a common collection is relatively large, e.g. 100 or 200, the computation of these matrices is nearly impossible. Even if they could be estimated, the disk space required to store them would be huge. Subsequently, alternative solutions must be looked for to overcome this drawback.

Our proposal to solve this problem is based on storing, instead of the conditional probability matrix $p(D_j|T_1,\ldots,T_k)$ on each document node, a *probability function*, in such a way that given a configuration of values of the parent variables, as well as a value for the document variable, the function is evaluated just at the moment in which the propagation process requires a specific conditional probability value (i.e., we use an implicit representation of the probability values instead of a explicit one).

Given a document D_j, the set of terms which have been assigned to it, $\Pi(D_j) = \{T_1,\ldots,T_k\}$, and a particular configuration of $\Pi(D_j)$, $\pi(D_j)$, we are going to use formulas that allow us to compute the conditional probability $p(d_j|\pi(D_j)) = p(d_j|t_1,\ldots,t_k)$ as a function of the individual probabilities $p(d_j|t_i)$, $i = 1,\ldots,k$. The following three functions have been designed to be applied to our model:

- *fp1*: It can be considered as a normalized idf representing that the more a term occurs in the collection, the less probable the documents which contain it are relevant to that term.

$$p(d_j|t_i) = \frac{idf_i}{log(N+1)log(k+1)} \quad (1)$$

where k is the number of terms in document D_j and $p(d_j|\bar{t}_i) = \alpha$, α being a number close to 0.0. The factor $log(k+1)$ represents the idea that for a document indexed by many terms, to know that only one term is relevant gives less information than when the document has few terms.

The aggregation function is based on an "Or-gate" [14] and is the following: let $p(d_j|t_i) = p_i$ and $p(d_j|\bar{t}_i) = q_i$, then we will show the expressions when all the terms are relevant, not relevant, and the case in which there are some of them that are relevant, and the rest not relevant:

$$p(d_j|t_1,\ldots,t_k) = 1 - \prod_{i=1}^{k}(1-p_i), \quad (2)$$

$$p(d_j|\bar{t}_1,\ldots,\bar{t}_k) = \prod_{i=1}^{k} q_i, \quad (3)$$

and finally,

$$p(d_j|t_1,\ldots,t_h,\bar{t}_{h+1},\ldots,\bar{t}_k) = 1 - \prod_{i=1}^{h}(1-p_i) + \prod_{i=h+1}^{k} q_i \prod_{i=1}^{h}(1-p_i) \quad (4)$$

- *fp2*: Represents a normalized sum of the tf.idf values from those terms which are relevant in the configuration. That is to say,

$$p(d_j|\pi(D_j)) = \frac{\sum_{i=1}^{k} p_i}{\sum_{i=1}^{k} tf_{ji}idf_i}, \quad (5)$$

p_i being $tf_{ji}idf_i$ if $T_i = t_i$, and 0 when $T_i = \bar{t}_i$.

- *fp3*: This third way of computing conditional probabilities represents the proportion of relevant terms in the configuration from the total number of terms in the document. In this case, the probability of relevance for a document is calculated as:

$$p(d_j|\pi(D_j)) = \frac{\sum_{i=1}^{k} p_i}{k} \quad (6)$$

with p_i being equals to 1 if $T_i = t_i$, and 0 when $T_i = \bar{t}_i$.

Both Bayesian network models, the simple and the augmented, will use this kind of probability functions instead of matrices.

In their model, Ribeiro and Muntz [15] also use a probability function, more precisely the cosine similarity formula [17], and alternatively, Indrawan et al. [12] use a kind of clustering, grouping the terms of a document by inserting intermediate nodes to which each group is connected. These new nodes are, at the same time, linked to the corresponding document node, thus reducing the size of the conditional probability matrices.

As mentioned at the beginning of this subsection, this problem with conditional probability matrices is also present in term nodes when the learning algorithm assigns too much parents to a term, and was solved in the learning phase by limiting the number of parents of a node [2].

4.3 Computation of conditional probabilities

Let us now concentrate on the computation of the conditional probability matrices that are going to be stored in the term nodes. For instance, given a node T_i and its parents, $\Pi(T_i)$, we estimate the conditional probability values by means of the maximum likelihood estimator, i.e.

$$p(t_i|\pi(T_i)) = \frac{n(t_i,\pi(T_i))}{n(\pi(T_i))} \quad (7)$$

$n(t_i,\pi(T_i))$ being the number of cases in the collection that both appear, the term T_i and those terms in $\Pi(T_i)$ that take the value relevant in the configuration $\pi(T_i)$. Similarly, $n(\pi(T_i))$ represents the number of cases where all the variables in $\pi(T_i)$ being relevant appear in a document. Once matrices were estimated, we observed that they had a lot of probabilities equal to 0.0 or very close to it (similarly, there were others very close to 1.0); this fact leads us immediately to think that the data are very disperse, and therefore there were several co-occurrence patterns whose frequency is 0. To improve the matrix quality, we decided 'softening' or 'relaxing' the probabilities using two different Bayesian methods [11]:

- $s1$:

$$p(t_i|\pi(T_i)) = \frac{n(t_i,\pi(T_i)) + 1}{n(\pi(T_i)) + \mid T_i \mid} \quad (8)$$

where $\mid T_i \mid$ is the number of values that T_i can have, i.e., 2.

- $s2$:

$$p(t_i|\pi(T_i)) = \frac{n(t_i,\pi(T_i)) + s\frac{n(t_i)}{N}}{n(\pi(T_i)) + s} \quad (9)$$

s being a parameter (that can be interpreted as the sample size used to estimate the a priori distribution) and N is the

total number of data, i.e., the number of documents in the collection.

For $s2$, two variations have been considered:

- *Global relaxing* (**g**): the expression (9) is applied to all the components of the conditional probability matrices.
- *Selective relaxing* (**s**): the expression (9) is only applied to those configurations of the conditioning variables whose values are all of them relevant. The reason why we do this type of relaxing is to establish a greater distance with the rest of probabilities, highlighting the importance of these two probabilities in which all the parents are relevant with respect to the others in the matrix.

5 Experimentation

In the following subsections we will briefly describe the material used in our experiments (databases and software), as well as the experimental design and the obtained results.

5.1 Standard test collection

In IR it is very common to test the IR systems developed over standard collections, with the aim of comparing the performance obtained by these different IR models. These collections are composed of a set of documents, a group of queries, and a set of relevance judgments, indicating which documents are relevant for each query.

In our experiments, we have used the Adi collection, obtained from the Computer Science Department ftp site at Cornell University [4], whose subject is Information Science. This collection consist on 82 documents, 828 terms and 35 queries.

As can be noticed, the collection size is relatively small and could not represent the sizes of actual collections, but the reason why we chose this database to carry out our experiments is that it establishes a good test bank to experiment with our model in a very short time, preparing and tuning our algorithms to work with bigger collections.

5.2 The retrieval process. Experiment design

Given a query submitted to our system, the retrieval process starts placing the evidences, i.e., the terms belonging to the query, in the term subnetwork by setting their states to *'the term is relevant'*. The propagation process is run, obtaining for each document its probability of relevance given that the terms in the query are also relevant. Then, the documents are sorted by its posterior probability to carry out the evaluation process.

Model	Prior prob.	Cond. prob.	Prob. functions
SIRBN	pp1, pp2, pp3	–	fp1, fp2, fp3
AIRBN	pp3	s1, s2	fp1, fp2

Table 1. : Experiment design.

The propagation method that we have used is the so called *'Importance Sampling'* [13], which is a new Monte-Carlo propagation algorithm integrated in the *Elvira System* [6]. We choose an approximate propagation method due to the size of the probability distributions stored in document nodes. An exact method would need to evaluate all the possible configurations for the parents of each document node, which represents a high time consuming task.

The experiments that we have designed to test our IR models, are divided into two groups, according to the two models introduced in this paper. We present in Table 1 the main different parameters which have been used in the experimentation for both, the Simple and Augmented Information Retrieval Bayesian Network, SIRBN and AIRBN, respectively. In order to reduce the number of experiment for AIRBN, and considering that the number of root nodes has been drastically reduced in the polytree, we only use the $pp3$ prior probability distribution that takes into account the quality of the term.

For $s2$ and for its two variations, we have to specify the parameter s for equation (9), which will regulate the relaxation. We have chosen the values 3, 5, 7 and 8. Summing up, we have designed 33 experiments, 9 belonging to the simple model, and 24 to the augmented model. In the following subsection we are going to introduce the results we have obtained in our experimentation.

5.3 Results

The evaluation of retrieval performance is basically carried out by means of two complementary measures: *recall* and *precision* estimations [16, 17]. The first one measures the ability of the IRS to present all the relevant documents (number of relevant document retrieved / number of relevant documents). The second one, precision, measures its ability to present only the relevant documents (number of relevant documents retrieved / number of documents retrieved).

To establish the quality of our results, we will compare them to the well known IR system Smart [4], developed by G. Salton and C. Buckley [1]. Smart implements the vector space model, in which queries and documents are represented by vectors containing weights. In our comparisons, we have used the 'ntn' weighting scheme, based on the $tf.idf$ weight. In Tables 2 and 3 we present for each experiment the following data: number of Relevant Retrieved documents (R) – we use the default number of retrieved doc-

Exp.	R	Exp.	ER	Exp.	EP
Smart	91	**Smart**	0.5964	**Smart**	0.1733
fp1, pp3	87	fp2, pp2	0.5909	fp1, pp2	0.1657
fp1, pp2	87	fp1, pp2	0.5787	fp1, pp3	0.1657
fp2, pp2	87	fp1, pp3	0.5587	fp2, pp2	0.1638
fp3, pp2	86	fp2, pp3	0.5494	fp3, pp2	0.1562
fp2, pp3	82	fp3, pp2	0.5344	fp2, pp3	0.1543
fp2, pp1	79	fp2, pp1	0.5275	fp2, pp1	0.1505
fp3, pp3	79	fp3, pp3	0.4891	fp3, pp3	0.1505
fp3, pp1	69	fp3, pp1	0.4430	fp3, pp1	0.1314
fp1, pp1	34	fp1, pp1	0.1863	fp1, pp1	0.064

Table 2. Experimentation results for the Simple IR Bayesian Network.

uments established in Smart, i.e., 15–, the Exact Recall (*ER*) and the Exact Precision (*EP*). For illustrative purpose, the experimental results have been sorted in both tables by its effectiveness measures, i.e., R, ER and EP. These tables also include the results obtained with Smart.

As it could be expected, from both tables, we can conclude that the performance of the retrieval system is highly dependent on the network structure and also on the probability distributions stored in the nodes, being the augmented network preferable to the simple one.

Focusing on the results for SIRBN, it can be concluded that it is useful to assign small prior probability of relevance for the terms in the collection, like $pp2$ and $pp3$ do. Moreover, considering the conditional probability distributions stored in document nodes, is preferable to incorporate information related with the relevance or specificity of the terms in the collection, as $fp1$ and $fp2$ do. So that, we do not consider the probability function $fp3$ in the experiment with the augmented network.

Now, we are going to analyze the results obtained with AIRBN (see Table 3). Similarly with the simple model, it is difficult to discern which probability function, $fp1$ or $fp2$, is the best one. In this case, two main discussion ought to be done: On the one hand, considering $fp2$, it is better to use $s1$ as relaxing criterion. When we use $s2$, it can be found that, in general, selective relaxing has better performance than global relaxing. With respect to the parameter s used in $s2$ it seems to be necessary a tuning process, being preferable small values. On the other hand, considering $fp1$, we obtain the worst results using $s1$. We think that the reason is in the way in which $s1$ increments the probability of relevance when we are conditioning to a non relevant term. This effect is reconsidered when using $s2$, obtaining better results. Again, using the selective relaxing seems to be better for improving the results. Nevertheless, a good tuning of the relaxing parameter seems to be important for getting good results.

Exp.	R	Exp.	ER	Exp.	EP
s2,s,7,fp1	96	s2,g,3,fp2	0.6356	s2,s,7,fp1	0.1829
s2,g,5,fp1	95	s2,s,3,fp2	0.6284	s2,s,9,fp1	0.181
s2,s,9,fp1	95	s2,s,7,fp1	0.6141	s2,g,5,fp1	0.181
s1,fp2	94	s2,s,8,fp2	0.6139	s2,g,3,fp2	0.179
s2,g,3,fp2	94	fp2	0.6136	s1,fp2	0.179
s2,s,3,fp2	93	s2,s,5,fp2	0.6106	s2,s,8,fp2	0.1771
s2,s,5,fp1	93	s2,s,7,fp2	0.6085	s2,s,5,fp1	0.1771
s2,s,8,fp2	93	s1,fp2	0.6064	s2,s,3,fp2	0.1771
fp2	92	s2,s,9,fp1	0.6053	s2,g,8,fp1	0.1752
s2,g,8,fp2	92	s2,s,9,fp2	0.6035	s2,g,3,fp1	0.1752
s2,s,3,fp1	92	s2,s,5,fp1	0.6035	s2,s,3,fp1	0.1752
s2,g,3,fp1	92	s2,g,5,fp1	0.6034	fp2	0.1752
s2,s,7,fp2	91	**Smart**	0.5964	s2,s,5,fp2	0.1733
s2,s,5,fp2	91	s2,s,3,fp1	0.5961	s2,s,7,fp2	0.1733
s2,s,8,fp1	91	s2,g,3,fp1	0.594	s2,s,8,fp1	0.1733
Smart	91	s2,g,9,fp1	0.5854	**Smart**	0.1733
s2,g,8,fp1	91	s2,g,8,fp1	0.5831	s2,g,7,fp1	0.1714
s2,g,7,fp1	90	s2,g,7,fp2	0.5803	s2,s,9,fp2	0.1714
s2,g,9,fp1	90	s2,s,8,fp1	0.5794	s2,g,9,fp1	0.1714
s2,s,9,fp2	90	s2,g,7,fp1	0.5792	s2,g,8,fp2	0.1714
s2,g,7,fp2	87	s2,g,9,fp2	0.5724	s2,g,7,fp2	0.1657
fp1	86	s2,g,5,fp2	0.5718	s2,g,5,fp2	0.1638
s2,g,5,fp2	86	fp1	0.5628	fp1	0.1638
s2,g,9,fp2	86	s2,g,8,fp2	0.5616	s2,g,9,fp2	0.1638
s1,fp1	82	s1,fp1	0.5482	s1,fp1	0.1562

Table 3. Experimentation results for the Augmented IR Bayesian Network.

6 Concluding remarks and future work

In this paper we have introduced a new Bayesian network-based model for IR, which has been based on a thesaurus that captures the most important term relationships and is represented by a polytree structure. We have shown how the retrieval effectiveness greatly depends on the quality of the term subnetwork, as well as on other parameters as probability functions and the type of relaxation in the conditional probability matrices.

The results obtained from our experiments show how the augmented IR Bayesian network improves in certain cases those obtained by Smart, but conveniently tuning the different parameters.

The main conclusions that we can obtain from this work are the following:

- The prior probability matrices are very relevant to the simple IR Bayesian network, but this network is not good enough to get better results.
- The results are improved when the term relationships

represented in the term subnetwork are closer to the relationships present in the real model.

- The relaxing of probability matrices is shown to be a good technique to improve the retrieval effectiveness, although is very relevant the factor used to soften these probabilities, because an over-relaxation could lead to decrease the IRS performance.

- The alternative of using probability functions in the document nodes instead of conditional probability matrices is one of the most suitable solutions to the problem of the matrix sizes, although the design process of these functions requires special care.

Our future works on this field are going to be focused on the following:

- Proving the goodness of our model applying it on different standard document collections, obtaining the set of values for all the parameters in which the retrieval effectiveness is the highest achievable.

- Designing new and more accurate probability functions.

- Improving the quality of the term subnetwork in order to learn a more accurate model. To carry out this task, we are planning to develop efficient learning algorithms to:

    1. Learn an unique term subnetwork but more complex than a polytree, and closer to the real model. In this case we have to take care because, as mentioned, the more complex the network is, the more time to learn and propagate is consumed, and
    2. apply clustering techniques to learn several and smaller term subnetworks, but, and due to the reduced sizes, with a greater complexity.

- Learning of more accurate document networks, taking advantage of the relationships between the documents to improve the retrieval effectiveness.

- Designing of appropriate propagation methods to be completely adapted to the networks we will use, with the aim of reducing the propagation time.

Acknowledgments This work has been supported by the Spanish Comisión Interministerial de Ciencia y Tenología (CICYT) under Project n. TIC96-0781.

References

[1] C. Buckley. Implementation of the SMART Information Retrieval System. Dpt. Computer Science. Cornell University. Tr-85-686 (1985).

[2] L. M. de Campos, J. M. Fernández, J. F. Huete. Query expansion in Information Retrieval Systems using a Bayesian Network-based Thesaurus. Proc.14th Uncertainty in Artificial Intelligence Conf., G. Cooper and S. Moral ed., 53 - 60 (Morgan Kaufman, 1.998).

[3] C. K. Chow, C. N. Liu, Approximating discrete probability distributions with dependence trees, IEEE Transaction on Information Theory, 14 462-467 (1968).

[4] Computer Science Department. Cornell University. ftp.cs.cornell.edu

[5] W. B. Croft, H. Turtle, Text Retrieval and Inference, Text-based Intelligent Systems, Paul Jacobs Ed., 127-156 (Lawrence Erlbaum, 1992).

[6] Dpto. Ciencias de la Computación. Universidad de Granada. http://leo.ugr.es/elvira/elvira.html

[7] N. Fuhr, Models for retrieval with probabilistic indexing, Information Processing and Management, 25(1) 55-72 (1989).

[8] N. Fuhr, From uncentain inference to agent-based Information Retrieval, VDI'98 - Arbeitstagung der GI-Fachgruppen verteilte knstliche intelligenz, databanken und information retrieval (1998).

[9] R. Fung, B. Del Favero, Applying Bayesian networks to information retrieval, Communications of the ACM, 38(3) 42-57 (1995).

[10] D. Ghazfan, M. Indrawan, B. Srinivasan, Towards meaningful Bayesian networks for information retrieval systems, Proceedings of the IPMU'96 Conference, 841-846 (1996).

[11] I.J. Good, The Estimation of Probabilities (MIT Press, Cambridge, 1965).

[12] M. Indrawan, D. Ghazfa, B. Srinivasan. Using Bayesian networks as retrieval engines. Proceedings of the 6th Text Retrieval Conference. NIST Special Publication 500-238, 437-444 (1996).

[13] A. Salmerón, A. Cano, S. Moral. Importance Sampling in Bayesian Networks Using Probability Trees. #DECSAI-98-02-20 (1998).

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

[15] B. Ribeiro, R. Muntz. A belief network model for IR. Proc. 19th ACM-SIGIR Conference (1996).

[16] C. J. van Rijsbergen, Information Retrieval, 2nd. edition (Butterworths, 1979).

[17] G. Salton, J. McGill, Introduction to modern information retrieval (McGraw Hill, 1993).

[18] H. Turtle, Inference networks for document retrieval, Ph.D. Dissertation, Computer and Information Science Dpt., University of Massachusetts (1991).