Inference Networks for Document Retrieval

Howard Turtle and W. Bruce Croft
Computer and Information Science Department
University of Massachusetts
Amherst, MA 01003

Abstract

The use of inference networks to support document retrieval is introduced. A network-based retrieval model is described and compared to conventional probabilistic and Boolean models.

1 Introduction

Network representations have been used in information retrieval since at least the early 1960's. Networks have been used to support diverse retrieval functions, including browsing [TC89], document clustering [Cro80], spreading activation search [CK87], support for multiple search strategies [CT87], and representation of user knowledge [OPC86] or document content [TS85].

Recent work suggests that significant improvements in retrieval performance will require techniques that, in some sense, "understand" the content of documents and queries [vR86, Cro87] and can be used to infer probable relationships between documents and queries. In this view, information retrieval is an inference or evidential reasoning process in which we estimate the probability that a user's information need, expressed as one or more queries, is met given a document as "evidence." Network representations show promise as mechanisms for inferring these kinds of relationships [CT89,CK87].

The idea that retrieval is an inference or evidential reasoning process is not new. Cooper's logical relevance [Coo71] is based on deductive relationships between representations of documents and information needs. Wilson's situational relevance [Wil73] extends this notion to incorporate inductive or uncertain inference based on the degree to which documents support information needs. The techniques required to support these kinds of inference are similar to those used in expert systems that must reason with uncertain information. A number of competing inference models have been developed for these kinds of expert systems [KL86,LK88] and several of these models can be adapted to the document retrieval task.

In the research described here we adapt an inference network model to the retrieval task. The use of the model is intended to:

- Support the use of multiple document representation schemes. Research has shown that a given query will retrieve different documents when applied to different repre-

sentations, even when the average retrieval performance achieved with each representation is the same. Katzer, for example, found little overlap in documents retrieved using seven different representations, but found that documents retrieved by multiple representations were likely to be relevant [KMT+82]. Similar results have been obtained when comparing term- with cluster-based representations [CH79] and term-with citation-based representations [FNL88].

- Allow results from different queries and query types to be combined. Given a single natural language description of an information need, different searchers will formulate different queries to represent that need and will retrieve different documents, even when average performance is the same for each searcher [MKN79,KMT+82]. Again, documents retrieved by multiple searchers are more likely to be relevant. A description of an information need can be used to generate several query representations (e.g., probabilistic, Boolean), each using a different query strategy and each capturing different aspects of the information need. These different search strategies are known to retrieve different documents for the same underlying information need [Cro87].

- Facilitate flexible matching between the terms or concepts mentioned in queries and those assigned to documents. The poor match between the vocabulary used to express queries and the vocabulary used to represent documents appears to be a major cause of poor recall [FLGD87]. Recall can be improved using domain knowledge to match query and representation concepts without significantly degrading precision.

The resulting formal retrieval model integrates several previous models in a single theoretical framework; multiple document and query representations are treated as evidence which is combined to estimate the probability that a document satisfies a user's information need.

In what follows we briefly review candidate inference models, present an inference network-based retrieval model, and compare the network model to current retrieval models.

2 Inference networks

The development of automated inference techniques that accommodate uncertainty has been an area of active research in the artificial intelligence community, particularly in the context of expert systems [KL86,LK88]. Popular approaches include those based on purely symbolic reasoning [Coh85,Doy79], fuzzy sets [Zad83], and a variety of probability models [Nil86,Che88]. Two inference models based on probabilistic methods are of particular interest: Bayesian inference networks [Pea88,LS88] and the Dempster-Shafer theory of evidence [Dem68,Sha76].

A Bayesian inference network is a directed, acyclic dependency graph (DAG) in which nodes represent propositional variables or constants and edges represent dependence relations between propositions. If a proposition represented by a node p "causes" or implies the proposition represented by node q, we draw a directed edge from p to q. The node q contains a *link* matrix that specifies $P(q|p)$ for all possible values of the two variables. When a node has multiple parents, the link matrix specifies the dependence of that node on the set of parents (π_q) and characterizes the dependence relationship between that node

2

and all nodes representing its potential causes.[1] Given a set of prior probabilities for the roots of the DAG, these networks can be used to compute the probability or degree of belief associated with all remaining nodes.

Different restrictions on the topology of the network and assumptions about the way in which the connected nodes interact lead to different schemes for combining probabilities. In general, these schemes have two components which operate independently: a *predictive* component in which parent nodes provide support for their children (the degree to which we believe a proposition depends on the degree to which we believe the propositions that might cause it), and a *diagnostic* component in which children provide support for their parents (if our belief in a proposition increases or decreases, so does our belief in its potential causes). The propagation of probabilities through the net can be done using information passed between adjacent nodes.

The Dempster-Shafer theory of evidence, although not originally cast as a network model, can be used as an alternative method for evaluating these kinds of probabilistic inference networks. Rather than computing the belief associated with a query given a set of evidence, we can view Dempster-Shafer as computing the probability that the evidence would allow us to prove the query. The degree of support parameters associated with the arcs joining nodes are not interpreted as conditional probabilities, but as assertions that the parent node provides support for the child (is *active*) for some proportion p of the time and does not support the child for the remainder of the time. For an *and*-combination we compute the proportion of the time that all incoming arcs are active. For an *or*-combination we compute the proportion of the time that at least one parent node is active. To compute the provability of the query given a document, we examine all paths leading from the document to the query and compute the proportion of time that all of the arcs on at least one proof path are active. Given the structure of these networks, this computation can be done using series-parallel reduction of the subgraph joining the document and query in time proportional to the number of arcs in the subgraph.

The Bayesian and Dempster-Shafer models are different and can lead to different results. However, under the assumption of disjunctive rule interaction (so called "noisy-OR") and the interpretation of an arc from a to b as $P(b|a) = p$ and $P(b|\neg a) = 0$, the Bayesian and Dempster-Shafer models will produce similar results [Pea88, page 446]. The document retrieval inference networks described here are based on the Bayesian inference network model.

The use of Bayesian inference networks for information retrieval represents an extension of probability-based retrieval research dating from the early 1960's [MK60]. It has long been recognized that some terms in a collection are more significant than others and that information about the distribution of terms in a collection can be used to improve retrieval performance. The use of these networks generalizes existing probabilistic models and allows integration of several sources of knowledge in a single framework.

---

[1]While this probability specification is generally referred to as a link matrix, it is actually a tensor.
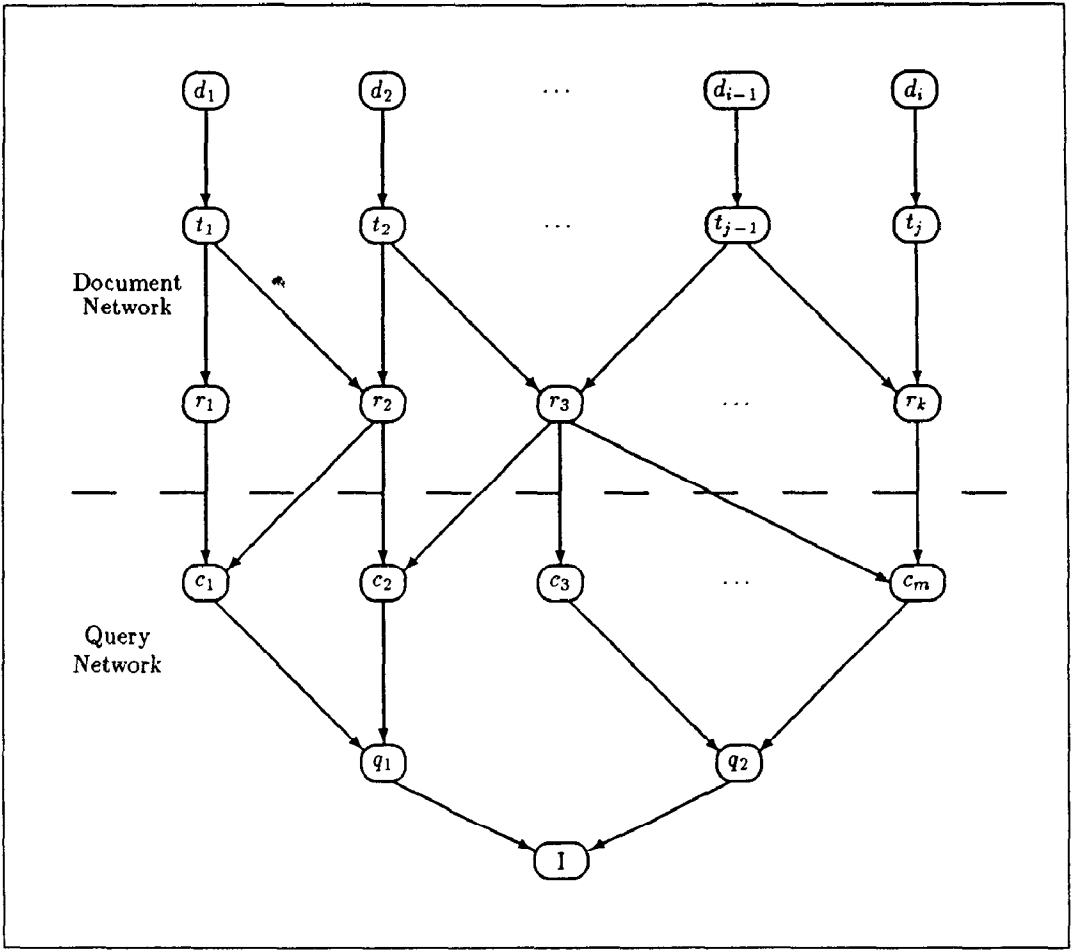
Figure 1: Basic document inference network

3   Basic Model

The basic document retrieval inference network, shown in Figure 1, consists of two component networks: a document network and a query network. The document network represents the document collection using a variety of document representation schemes. The document network is built once for a given collection and its structure does not change during query processing. The query network consists of a single node which represents the user's information need and one or more query representations which express that information need. A query network is built for each information need and is modified during query processing as existing queries are refined or new queries are added in an attempt to better characterize the information need. The document and query networks are joined by links between representation concepts and query concepts. All nodes in the inference network take on values

from the set {*false,true*}.

3.1   Document network

The document network consists of document nodes (d_i's), text representation nodes (t_j's), and concept representation nodes (r_k's). Each document node represents a document in the collection. A document node corresponds to the event that a specific document has been observed. The form of the document represented depends on the collection and its intended use, but we will assume that a document is a well defined object and will focus on traditional document types (e.g., monographs, journal articles, office documents).

Document nodes correspond to abstract documents rather than their physical representations. A text representation node or text node corresponds to a specific text representation of a document. A text node corresponds to the event that a text representation has been observed. We focus here on the text content of documents, but the network model can support documents nodes with multiple children representing additional component types (e.g., figures, audio, or video). Similarly, a single text might be shared by more than one document. While shared components is rare in traditional collections (an example would be a journal article that appears in both a serial issue and in a reprint collection) and is not generally represented in current retrieval models, it is common in hypertext systems. For clarity, we will consider only text representations and will assume a one-to-one correspondence between documents and texts. The dependence of a text upon the document is represented in the network by an arc from the document node to the text node.

The content representation nodes or representation nodes can be divided into several subsets, each corresponding to a single representation technique that has been applied to the document texts. For example, if a collection has been indexed using automatic phrase extraction and manually assigned index terms, then the set of representation nodes will consist of two distinct subsets or content representation types with disjoint domains. Thus, if the phrase "information retrieval" has been extracted and "information retrieval" has been manually assigned as an index term, then two representation nodes with distinct meanings will be created. One corresponds to the event that "information retrieval" has been automatically extracted from a subset of the collection, the second corresponds to the event that "information retrieval" has been manually assigned to a (presumably distinct) subset of the collection. We represent the assignment of a specific representation concept to a document by a directed arc to the representation node from each text node corresponding to a document to which the concept has been assigned. For now we assume that the presence or absence of a link corresponds to a binary assigned/not assigned distinction, that is, there are no partial or weighted assignments.

In principle, the number of representation schemes is unlimited; in addition to phrase extraction and manually assigned terms we would expect representations based on natural language processing and automatic keyword extraction. For any real document collection, however, the number of representations used will be fixed and relatively small. The potential domain of each representation scheme may also be unlimited, but the actual number of primitive representation concepts defined for a given collection is fixed by the collection. The domain for most automated representation schemes is generally bounded by some

function of the collection size (e.g., the number of keywords cannot exceed the number of words in a collection). For manual representation schemes the domain size is limited by the number of documents, the representation scheme itself (e.g., a controlled vocabulary), and the amount of time a human expert can spend analyzing each document.

The basic document network shown in Figure 1 is a simple three level DAG in which document nodes are roots, text nodes are interior nodes, and representation nodes are leaves. Document nodes have exactly one text node as a child and each text node has one or more representation nodes as children.

Each document node has a prior probability associated with it that describes the probability of observing that document; this prior probability will generally be set to 1/(collection size) and will be small for real collections. Each text node contains a specification of its dependence upon its parent; by assumption, this dependence is complete, a text node is observed ($t_i = true$) exactly when its parent document is observed ($d_i = true$).

Each representation node contains a specification of the conditional probability associated with the node given its set of parent text nodes. This specification incorporates the effect of any indexing weights (e.g., term frequency for each parent text) or term weights (e.g., inverse document frequency) associated with the representation concept. While, in principle, this would require $O(2^n)$ space for a node with n parents, in practice we use canonical representations that allow us to compute the required conditional probabilities when needed. These canonical schemes require $O(n)$ space if we weight the contribution of each parent or $O(1)$ space if parents are to be treated uniformly.

3.2 Query network

The query network is an "inverted" DAG with a single leaf that corresponds to the event that an information need is met and multiple roots that correspond to the concepts that express the information need. As shown in Figure 1, a set of intermediate query nodes may be used when multiple queries express the information need. These nodes are a representation convenience; it is always possible to eliminate them by increasing the complexity of the distribution specified at the node representing the information need.

In general, the user's information need is internal to the user and is not precisely understood. We attempt to make the meaning of an information need explicit by expressing it in the form of one or more queries that have formal interpretations. These queries may be generated from a single natural language description (e.g., keywords or phrases for a probabilistic search, a Boolean representation, sample documents, ...) or they may represent additional sources of information (e.g., an intermediary's description of the user or of the information need, or feedback provided by the user). It is unlikely that any of these queries will correspond precisely to the information need, but some will better characterize the information need than others and several query specifications taken together may be a better representation than any of the individual queries.

The roots of the query network are query concepts; they correspond to the primitive concepts used to express the information need. A single query concept node may have several representation concept nodes as parents. Each query concept node contains a specification of its dependence on the set of parent representation concepts. The query concept nodes

define the mapping between the concepts used to represent the document collection and the concepts used in the queries. In the simplest case, the query concepts are the same as the representation concepts so each query concept has exactly one parent. In a slightly more complex example, the query concept "information retrieval" may have as parents both the node corresponding to "information retrieval" as a phrase and the node corresponding to "information retrieval" as a manually assigned term. As we add content representations to the document network and allow query concepts that do not explicitly appear in any document representation, the number of parents associated with a single query concept will increase.

A query concept is similar to a representation concept that is derived from other representation concepts (see section 5 for a discussion of derived representation concepts) and in some cases it will be useful to "promote" a query concept to a representation concept. For example, suppose that a researcher is looking for information on a recently developed process that is unlikely to be explicitly identified in any existing representation scheme. The researcher, if sufficiently motivated, could work with the retrieval system to describe how this new concept might be inferred from other representation concepts. If this new concept definition is of general interest, it can be added to the collection of representation concepts. This use of inference to define new concepts is similar to that used in RUBRIC [TS85].

The attachment of the query concept nodes to the document network has no effect on the basic structure of the document network. None of the existing links need change and none of the conditional probability specifications stored in the nodes are modified.

A query node represents a distinct query form and corresponds to the event that the query is satisfied. Each query node contains a specification of the dependence of the query on its parent query concepts. The link matrices that describe these conditional probabilities are discussed further in section 3.4, but we note that the form of the link matrix is determined by the query type; a link matrix simulating a Boolean operator is different than a matrix simulating a probabilistic or weighted query.

The single leaf representing the information need corresponds to the event that an information need is met. In general, we cannot predict with certainty whether a user's information need will be met by a document collection. The query network is intended to capture the way in which meeting the user's information need depends on documents and their representations. Moreover, the query network is intended to allow us to combine information from multiple document representations and to combine queries of different types to form a single, formally justified estimate of the probability that the user's information need is met. If the inference network correctly characterizes the dependence of the information need on the collection, the computed probability provides a good estimate.

3.3   Use of the inference network

The retrieval inference network is intended to capture all of the significant probabilistic dependencies among the variables represented by nodes in the document and query networks. Given the prior probabilities associated with the documents (roots) and the conditional probabilities associated with the interior nodes, we can compute the posterior probability or belief associated with each node in the network. Further, if the value of any variable

represented in the network becomes known we can use the network to recompute the probabilities associated with all remaining nodes based on this "evidence."

The network, taken as a whole, represents the dependence of a user's information need on the documents in a collection where the dependence is mediated by document and query representations. When the query network is first built and attached to the document network we compute the belief associated with each node in the query network. The initial value at the node representing the information need is the probability that the information need is met given that no specific document in the collection has been observed and all documents are equally likely (or unlikely). If we now observe a single document d_i and attach evidence to the network asserting $d_i = true$ we can compute a new belief for every node in the network given $d_i = true$. In particular, we can compute the probability that the information need is met given that d_i has been observed in the collection. We can now remove this evidence and instead assert that some d_j, $i \neq j$ has been observed. By repeating this process we can compute the probability that the information need is met given each document in the collection and rank the documents accordingly.

In principle, we need not consider each document in isolation but could look for the subset of documents which produce the highest probability that the information need is met. While a general solution to this best-subset problem is intractable, in some cases good heuristic approximations are possible. Best-subset rankings have been considered in IR [Sti75], and similar problems arise in pattern recognition, medical diagnosis, and truth-maintenance systems. See [Pea88] for a discussion of the best-subset or belief revision problem in Bayesian networks. At present, we consider only documents in isolation because the approach is computationally simpler and because it allows comparison with earlier retrieval models that produce document rankings consistent with the Probability Ranking Principle [Rob77] in which documents are considered in isolation.

The document network is built once for a given collection. Given one or more queries representing an information need, we then build a query network that attempts to characterize the dependence of the information need on the collection. If the ranking produced by the initial query network is inadequate, we must add additional information to the query network or refine its structure to better characterize the meaning of the existing queries. This feedback process is quite similar to conventional relevance feedback.

3.4   Link matrix forms

For all non-root nodes in the inference network we must estimate the probability that a node takes on a value given any set of values for its parent nodes. If a node a has a set of parents $\pi_a = \{p_1, \ldots, p_n\}$, we must estimate $P(a|p_1, \ldots, p_n)$.

The most direct way to encode our estimate is as a link matrix. Since we are dealing with binary valued propositions, this matrix is of size 2×2^n for n parents and specifies the probability that a takes the value $a = true$ or $a = false$ for all combinations of parent values. The update procedures for Bayesian networks then use the probabilities provided by the set of parents to condition over the link matrix values to compute the predictive component of our belief in a or $P(a = true)$. Similarly, the link matrix is used to provide diagnostic information to the set of parents based on our belief in a. As mentioned earlier,

8

encoding our estimates in link matrix form is practical only for nodes with a small set of parents, so our estimation task has two parts: how do we estimate the dependence of a node on its set of parents and how do we encode these estimates in a usable form?

We will describe four canonical link matrix forms, three for the Boolean operators and a fourth for simple probabilistic retrieval. For illustration, we will assume that a node Q has three parents A, B, and C and that

$$P(A = \text{true}) = a, \quad P(B = \text{true}) = b, \quad P(C = \text{true}) = c.$$

For *or*-combinations, Q will be true when any of A, B, or C is true and false only when A, B, and C are all false. This suggests a link matrix of the form

$$L_{\text{or}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Using a closed form of the update procedures, we have

$$
\begin{aligned}
P(Q = true) &= (1-a)(1-b)c + (1-a)b(1-c) + (1-a)bc + a(1-b)(1-c) \\
&\quad + a(1-b)c + ab(1-c) + abc \\
&= 1 - (1-a)(1-b)(1-c)
\end{aligned}
$$

which is the familiar rule for disjunctive combination of events that are not known to be mutually exclusive. Similar matrix forms can be developed for *and* ($P(Q = true) = abc$) and *not* ($P(Q = true) = 1 - a$).

If we restrict the parent nodes for any of these logic operators to values 0 or 1 then Q must also have a value of 0 or 1. If we allow terms to take on weights in the range $[0, 1]$ and interpret these weights as the probability that the term has been assigned to a document text, then these inference networks provide a natural interpretation for Boolean retrieval with weighted indexing. The use of these canonical forms to simulate Boolean retrieval is discussed in section 4.3

For probabilistic retrieval each parent has a weight associated with it, as does the child. In this weighted-sum matrix, our belief in Q depends on the specific parents that are true – parents with larger weights have more influence in our belief. If we let $w_a, w_b, w_c \geq 0$ be the parent weights, $0 \leq w_q \leq 1$ the child weight, and $t = w_a + w_b + w_c$, then we have a link matrix of the form

$$
\begin{pmatrix}
1 & \frac{(w_a+w_b)w_q}{t} & \frac{(w_a+w_c)w_q}{t} & \frac{w_a w_q}{t} & \frac{(w_b+w_c)w_q}{t} & \frac{w_b w_q}{t} & \frac{w_c w_q}{t} & 1-w_q \\
0 & \frac{w_c w_q}{t} & \frac{w_b w_q}{t} & \frac{(w_b+w_c)w_q}{t} & \frac{w_a w_q}{t} & \frac{(w_a+w_c)w_q}{t} & \frac{(w_a+w_b)w_q}{t} & w_q
\end{pmatrix}.
$$

Evaluation of this link matrix form results in

$$P(Q = true) = \frac{(w_a a + w_b b + w_c c)w_q}{t}.$$

This link matrix can be used to implement a variety of weighting schemes, including the familiar term weighting schemes based on within-document term frequency (*tf*), inverse

document frequency (*idf*) or both (*tf.idf*). To illustrate a *tf.idf* weighting, let Q be a representation node and let A, B, and C be document nodes. Let w_a, w_b, and w_c be the normalized *tf* values for A, B, and C, let idf_q be the normalized *idf* weight for Q, and let

$$w_q = idf_q \cdot (w_a + w_b + w_c). \tag{1}$$

Given our basic model, when A is instantiated, belief in Q is given by

$$\begin{aligned}
\mathrm{bel}(Q) &= \frac{w_a w_q}{w_a + w_b + w_c} \\
&= \frac{tf_a \cdot idf_q \cdot (w_a + w_b + w_c)}{w_a + w_b + w_c} \\
&= tf_a \cdot idf_q
\end{aligned}$$

which is a form of *tf.idf* weight. In general, when a document is instantiated all representation concept nodes to which it is attached take on the *tf.idf* weight associated with the document/term pair.

The weight at Q has two distinct parts. The first part (idf_q in our example) acts to set the maximum belief achievable at a node. If, for some combination of parent values, our belief in Q is certain then this component disappears. Note that in this formulation, the *idf* component is dependent only upon the distribution of the term in the collection, not on the distribution of the term in relevant and non-relevant subsets. Relevance feedback is modeled as part of the query network and does not affect belief in representation concepts.

The second part ($w_a + w_b + w_c$ in our example) acts to normalize the parent weights. Equation 1 is appropriate for the basic model in which only one document is instantiated at a time. In the extended model of section 5 where multiple roots can be instantiated, this component is adjusted to normalize for the maximum achievable set of parent weights. In the general case, where all parents can take any value in the range $[0, 1]$, this normalizing component disappears.

These canonical forms are sufficient for the retrieval inference networks described here, but many others are possible (see section 4.3 for other examples). Further, when the number of parents is small (say, less than 5 or 6) we can use the full link matrix if the dependence of a node on its parents does not fit a canonical form.

4   Comparison with other retrieval models

The inference network retrieval model generalizes both the probabilistic and Boolean models. Inference networks can be used to simulate both probabilistic and Boolean queries and can be used to combine results from multiple queries.

In this section we compare the inference network model with probabilistic (sections 4.1 and 4.2) and Boolean (section 4.3) models and show how inference networks can be used to simulate both forms of retrieval. We then consider how the probabilities required by the model can be estimated (section 4.4); the estimation problems are essentially equivalent to those encountered with probabilistic or vector-space retrieval.
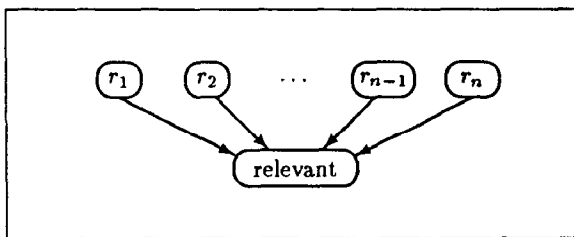
Figure 2: Inference network for binary independence model

4.1 Probabilistic retrieval models

Conventional probabilistic models [vR79,SM83] rank documents by the probability that each document would be judged relevant to a given query, $P(\text{relevant}|d_i)$.[2] This is, in many ways, similar to computing the probability that a user's information need is met given a specific document, $P(I|d_i)$. The principal differences between conventional probabilistic models and the model described here are: 1) most probabilistic models do not explicitly represent the query, 2) conventional probabilistic models do not distinguish between a document and its representations but treat a document as a single vector, and 3) the inference network model depends less upon Bayesian inversion than probabilistic models, Bayesian inversion is just one way to estimate $P(I|d_i)$ (or $P(Q|d_i)$ in the case of a single query).

In this section we summarize the major differences between the inference network and conventional probabilistic models by comparing the network model to the binary independence model. In the next section we provide a formal comparison of the inference network model with a recent probabilistic model that explicitly represents documents and queries.

An inference network that corresponds to the binary independence model [vR79] is shown in Figure 2. A document is represented by a vector whose components are indexing or representation concepts ($d_i = \{r_1, \ldots, r_n\}$). The set of concepts considered is generally restricted to the subset that actually occurs in the query. Comparing this network with that shown in Figure 1, we see that in the binary independence model, the document network is represented by a single level of representation nodes and the query network consists of a single relevance node. In order to implement this network we must somehow estimate the probability of relevance given the set of parent representation concepts and this estimate must incorporate all of our judgments about the probability that a representation concept should be assigned to a document, about the semantic and stochastic relationships between representation concepts, about the relationship between concepts named in the query and assigned to documents, and about the semantics of the query itself. This dependence is complex and its estimation is not a task we could expect users to perform willingly or

---

[2]Most probabilistic models do not actually compute $P(\text{relevant}|d_i)$, but simply rank documents using some function that is monotonic with $P(\text{relevant}|d_i)$. Like Fuhr ([Fuh89]), we believe that an estimate of the probability of relevance is more useful than the ranking by itself. A ranked list of documents in which the top ranked document has a probability of relevance of 0.5 should be viewed differently than a similar list in which the top ranked document has a probability of relevance of 0.95.

reliably.

One approach to simplifying the estimation task is to invoke Bayes' rule so that we need only estimate the probability that each representation concept occurs in relevant or non-relevant documents. This approach does not help to provide initial estimates of the probability distributions since these "simpler" estimates must still incorporate all of the judgments required for the "hard" estimate. The advantage of this approach is that, given samples of relevant and non-relevant documents, it is easy to compute $P(r_i)$ for the relevant sample and to use the result as an estimate of $P(r_i|\text{relevant} = true)$. We can use a similar estimate for $P(r_i|\text{relevant} = false)$. Given a set of independence assumptions and estimates for $P(d_i)$ and $P(\text{relevant} = true)$ we can compute $P(\text{relevant}|d_i)$.[3] Estimating $P(\text{relevant}|d_i)$ without the use of Bayes' rule would be extremely difficult.

Essentially the same procedures can be used to estimate $P(Q|d_i)$. The main difference between the two estimates is that instead of using the representation concepts directly we must compute $P(c_j|\pi_{c_j})$ and compute an expected value for $P(c_j|d_i)$ in order to estimate $P(Q|d_i)$.

The question remains, however, whether estimates of $P(\text{relevant}|d_i)$ or $P(Q|d_i)$ obtained in this way match users' intuition about the dependence. The fact that relevance feedback does improve retrieval performance suggests that the estimates of $P(\text{relevant}|d_i)$ do capture at least some of the dependence, but these estimates are generally based on a small number of relevant documents and are necessarily rather coarse.

While it is clear that estimating $P(\text{relevant}|d_i)$ directly from a small number of documents is impractical, it may be possible to obtain estimates of $P(Q|\pi_Q)$. Users may, for example, be able to assign importance to the concepts in their query and may be able to identify significant interactions between concepts. These estimates could improve the initial estimate and might be used in conjunction with the estimates derived from training samples.

A second approach to simplifying the estimation task is to identify the different types of judgments that enter into the overall estimate and to develop estimates for each type of judgment separately. The model presented here represents one decomposition in which the task of estimating the probability that a given document satisfies an information need consists of judgments about the relationship of a document to its text, the assignment of representation concepts to the text, the relationships between query and representation concepts, and the relationship between queries, query concepts, and the information need. Other decompositions are certainly possible and can be accommodated within the same general framework. The set of relationships presented here incorporates those judgments most important for current generation document retrieval systems.

When viewed this way, the probabilistic and inference models use two similar approaches to the same estimation problem. The probabilistic model uses a single, general purpose rule and makes assumptions about term dependence in order to estimate $P(\text{relevant}|d_i)$. The model presented here views the problem of estimating $P(I|d_i)$ as consisting of a set of logically related estimates. Each estimate is made independently using procedures specific to

---

[3] $P(d_i)$ and $P(\text{relevant} = true)$ do not play a major role in probabilistic models that only produce a document ranking but are required to compute $P(\text{relevant}|d_i)$.
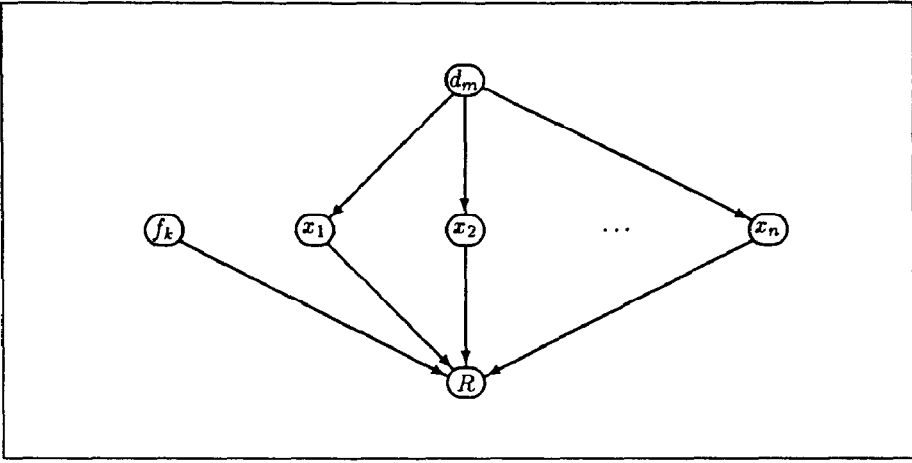
Figure 3: Inference network for the RPI model

the type of estimate; the "probabilistic" estimate of $P(Q|\pi_Q)$ is simply one component of the overall estimate. The component estimates are then combined in a manner consistent with the dependence relationships represented in the inference network to provide an estimate of $P(I|d_i)$.

4.2 Comparison with the RPI model

To further clarify the relationship between the inference network model and the probabilistic model, we will compare the inference network model with Fuhr's model for retrieval with probabilistic indexing (RPI model) [Fuh89]. To simplify the comparison, we will temporarily adopt Fuhr's notation. Let

d_m represent a document in the collection,

x   be the binary vector $(x_1, x_2, \ldots, x_n)$ in which each x_i

         corresponds to a document descriptor (representation concept),

f_k represent the query, and

R represent the event that a document is judged relevant to a query.

All variables are binary valued. In this model, $P(x_i = 1|d_m)$ is interpreted as the probability that a descriptor r_i is a "correct" indexing of d_m. Let X be the set of possible values for x, where $|X| \leq 2^n$.

The network shown in Figure 3 corresponds to the probability distribution

$$
\begin{aligned}
P(R, f_k, x_1, \ldots, x_n, d_m) &= P(R|f_k, d_m) \\
&= P(R|f_k, x_1, \ldots, x_n)P(x_1|d_m) \ldots P(x_n|d_m)P(f_k)P(d_m).
\end{aligned}
$$

We will evaluate this expression for a given document and query so f_k and d_m are known

and the distribution reduces to

$$P(R|f_k, d_m) = P(R|f_k, x_1, \ldots, x_n)P(x_1|d_m)\ldots P(x_n|d_m).$$

Assuming that the descriptors are assigned independently, that is

$$P(\mathbf{x}|d_m) = \prod_{1 \leq i \leq n} P(x_i|d_m),$$

the basic ranking expression for the network of Figure 3 is

$$P(R|f_k, d_m) = \sum_{\mathbf{x} \in X} P(R|f_k, \mathbf{x})P(\mathbf{x}|d_m). \tag{2}$$

Equation 2 is equivalent to the basic ranking expression used by Fuhr [Fuh89, equation 9]. Equation 2 can be expanded to the product form

$$P(R|f_k, d_m) = P(R|f_k) \prod_{1 \leq i \leq n} \left(\frac{p_{ik}}{q_i}u_{im} + \frac{1 - p_{ik}}{1 - q_i}(1 - u_{im}) \right) \tag{3}$$

where

$$\begin{aligned} p_{ik} &= P(x_i = 1|R, f_k) \\ q_i &= P(x_i = 1) \\ u_{im} &= P(x_i = 1|d_m). \end{aligned}$$

(Strictly speaking, the network corresponding to equation 2 should have a single node $\mathbf{x}$ in place of $x_1, \ldots, x_n$ since equation 2 makes no independence assumptions. Independence is, however, assumed in all derivations based on equation 2 so we have chosen to show it in the network.)

Using the same notation and variables, the network of Figure 1 can be reduced to the network of Figure 4. This inference network is described by the probability distribution

$$\begin{aligned} P(R, f_k, x_1, \ldots, x_n, d_m) &= P(R|d_m) \\ &= P(R|f_k)P(f_k|x_1, \ldots, x_n)P(x_1|d_m)\ldots P(x_n|d_m)P(d_m). \end{aligned}$$

Comparing Figure 4 with Figure 3 we see that in the inference network model the query does not appear as a separate prior (root) but is explicitly conditioned on the representation concepts. Again, d_m is given, so we have

$$P(R|d_m) = P(R|f_k)P(f_k|x_1, \ldots, x_n)P(x_1|d_m)\ldots P(x_n|d_m).$$

Applying Bayes' rule we get

$$P(R|d_m) = P(R|f_k)\frac{P(x_1, \ldots, x_n|f_k)P(f_k)}{P(x_1, \ldots, x_n)}P(x_1|d_m)\ldots P(x_n|d_m).$$
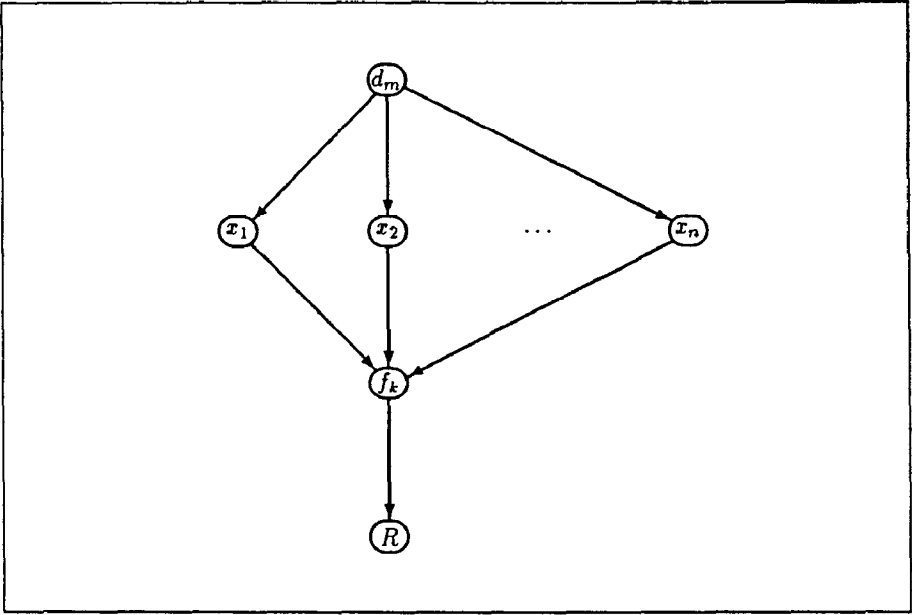
14

Figure 4: Example inference network

Assuming that the x_i are distributed independently in documents (4) and that the assignment of the x_i is independent of the query (5)

$$P(x_1,\ldots,x_n) = \prod_{1 \le i \le n} P(x_i) \qquad (4)$$

$$P(x_1,\ldots,x_n|f_k) = \prod_{1 \le i \le n} P(x_i|f_k) \qquad (5)$$

we have

$$P(R|d_m) = P(R|f_k)P(f_k) \sum_{X \in X} \prod_{1 \le i \le n} \frac{P(x_i|f_k)}{P(x_i)} P(x_i|d_m). \qquad (6)$$

The application of Bayes' rule essentially inverts the network of Figure 4 to obtain the equivalent network shown in Figure 5[4]. Note that the use of Bayes' rule here is to allow us to derive a closed-form ranking expression that can be compared with the RPI model. In practice, we would use an estimate of $P(f_k|x_1,\ldots,x_n)$ and would not invert the network.

---

[4]While the networks in Figures 4 and 5 are equivalent in the sense that the computed probability distributions are the same, Figure 5 does not lend itself to normal belief network updating procedures. In order to produce the new $P(x_i|f_k, d_m)$ link matrix and the new prior $P(f_k)$ we must make use of the assumed value of $P(d_m)$. In essence, when we invert the network we fold the prior probability of d_m into the new link matrix and extract a new prior for the query. This means that to test the effect of a change in $P(d_m)$, we would have to recompute the link matrices at each x_i and compute a new $P(f_k)$. With the network in Figure 4, we can change our assumed value for $P(d_m)$ without changing the probability information stored at each node.
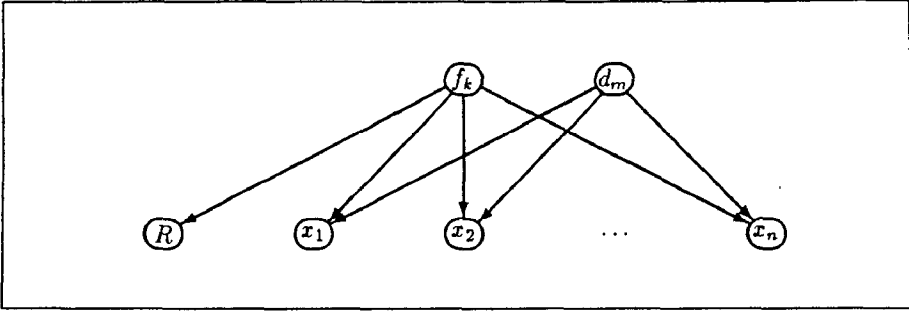
Figure 5: Effect of inversion

Equation 6 reduces to

$$P(R|d_m) = P(R|f_k)P(f_k) \prod_{1 \le i \le n} \left(\frac{P(x_i = 1|f_k)}{P(x_i = 1)} P(x_i = 1|d_m) \right.$$
$$\left. + \frac{P(x_i = 0|f_k)}{P(x_i = 0)} P(x_i = 0|d_m) \right).$$

If we let

$$
\begin{aligned}
p_{ik} &= P(x_i = 1|f_k) \\
q_i &= P(x_i = 1) \\
u_{im} &= P(x_i = 1|d_m)
\end{aligned}
$$

we get the ranking expression

$$P(R|d_m) = P(R|f_k)P(f_k) \prod_{1 \le i \le n} \left(\frac{p_{ik}}{q_i} u_{im} + \frac{1 - p_{ik}}{1 - q_i} (1 - u_{im}) \right) \qquad (7)$$

Equation 7 differs from equation 3 in that p_{ik} is conditioned only on the query and not on R and the resulting probability is normalized by $P(f_k)$. The difference in conditioning for p_{ik} arises because the network of Figure 4 implicitly assumes that $\mathbf{x}$ and R are conditionally independent given the query, that is, $\mathbf{x}$ cannot influence our assessment of relevance except through its effect on the query. The network of Figure 3 assumes that $\mathbf{x}$ and f_k are independent, but not necessarily conditionally independent given R, that is, $\mathbf{x}$ and the query can influence our assessment of relevance independently. Under the assumption of conditional independence

$$P(\mathbf{x}|R, f_k) = P(\mathbf{x}|f_k)$$

and the p_{ik} terms are identical. $P(f_k)$ is constant for a given query and does not affect the ranking so, under the assumption of conditional independence, the rankings produced by the two models are identical.

The networks in Figures 3 and 4 help to clarify the differences between the probabilistic and inference network retrieval models. In the network of Figure 3, the query is modeled

16

as a separate variable that is related to the possible document descriptions through the specification of $P(R|x, f_k)$. The network of Figure 4 explicitly models the dependence of the query on the document representation and the dependence of relevance on the query. Again, the network of Figure 4 asserts the independence of the document representation and relevance given the query; the document representation cannot influence the probability of relevance except through its influence on the query.

The principal difference between the two models, then, lies in the dependencies assumed. While we have chosen Fuhr's model as the basis for comparison, network forms could be developed for the many other probabilistic formulations. The chief advantage of the inference network model is that it allows complex dependencies to be represented in an easily understood form and it allows networks containing these dependencies to be evaluated without development of a closed form expression that captures these dependencies.

4.3 Boolean retrieval

Using the canonical link matrix forms of section 3.4 we can implement Boolean retrieval as follows. For clarity, we assume that the query and representation vocabularies are identical so we can omit query concepts from the network. We also assume that when one document is instantiated all remaining documents are set to false.

1. Use a canonical *or* matrix at each representation node. When a document is instantiated, all representation concepts to which it has been attached will have bel(r_i) = 1. All remaining representation concepts have bel(r_j) = 0.

2. Build an expression tree for the query. The root of the tree is the query and all arcs in the tree are directed toward the root. The leaves of this tree will be representation concepts and the interior nodes will correspond to expression operators. At each operator node use the canonical link matrix form for that operator. Attach this tree to the document network.

3. Using the evaluation procedure described in section 3.3, instantiate each document in turn and record the belief in the query node. Any document for which bel(Q) = 1 satisfies the query, any node for which bel(Q) < 1 does not.

Under the assumptions above and using binary indexing, bel(Q) can only have values 0 or 1 and the inference network simulates a conventional Boolean system exactly. If we relax the requirement that all uninstantiated documents be set to 0, then only documents for which bel(Q) = 1 satisfy the query and all remaining documents have a small but non-zero bel(Q).

The same probabilistic interpretation of the Boolean operators applies equally well to weighted indexing. Using the approach described in section 3.4 we can incorporate indexing weights by replacing the *or* link matrix at the representation concept nodes with a weighted-sum matrix incorporating the appropriate *tf* and *idf* weights. In this case, when a document is instantiated, all representation nodes to which it is attached take on the *tf.idf* weight for that term/document pair and all remaining representation nodes take on bel = 0. These weights are then combined using the closed-form expressions of section 3.4. In short, the

tf.idf weights are interpreted as probabilities and are combined using the normal rules for negation and for disjunctive or conjunctive combination of sets in an event space. As a result, the inference network model provides a natural interpretation of Boolean operations in probabilistic terms and of the meaning of indexing weights.

The binary nature of the retrieval decision in Boolean systems is frequently cited as a drawback [Cro86,SM83,Sal88]. We can relax our strict interpretation of the probabilistic semantics of the Boolean operators by allowing the number of parents=true to influence our belief. For example, we can choose a value $n \leq c \leq \infty$ and interpret the *and* operator to mean

$$
\begin{aligned}
P(Q_{and} = \text{true}|n \text{ parents} = \text{true}) &= 1 \\
P(Q_{and} = \text{true}|k \text{ parents} = \text{true}) &= 1 - \frac{n-k}{c}, \quad 0 < k < n \\
P(Q_{and} = \text{true}|\text{no parents} = \text{true}) &= 0
\end{aligned}
$$

and the *or* operator to mean

$$
\begin{aligned}
P(Q_{or} = \text{true}|n \text{ parents} = \text{true}) &= 1 \\
P(Q_{or} = \text{true}|k \text{ parents} = \text{true}) &= \frac{k}{c}, \quad 0 < k < n \\
P(Q_{or} = \text{true}|\text{no parents} = \text{true}) &= 0
\end{aligned}
$$

Since a node implementing the *not* operator has exactly one parent, its interpretation is unchanged. Under this interpretation, when $c = \infty$ the operators have their normal Boolean interpretation. As c decreases, our belief in Q depends increasingly on the number of parents that are true. When $c = n$ the distinction between *and* and *or* has disappeared and the link matrices for both operators are the same. The use of this parent weighting scheme is quite similar to the extended Boolean retrieval or *p*-norm model [Sal88,SM83]. The two approaches are equivalent when $c = n$ and $p = 1$ and when $c = p = \infty$; the resulting probability and similarity functions are monotonic for $n < c < \infty$ and $1 < p < \infty$.

4.4 Estimating the probabilities

Given the link matrix forms of section 3.4, we now consider the estimates required for the basic model of Figure 1. The only roots in Figure 1 are the document nodes; the prior probability associated with these nodes is set to 1/(collection size). Estimates are required for five different node types: text, representation and query concepts, query, and information need.

Text nodes. Since text nodes are completely dependent upon the parent document node, the estimate is straightforward. Since there is a single parent, a matrix form can be used; t_i is true exactly when d_i is true so

$$
L_{\text{text}} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.
$$

This matrix form is the inverse of that used for *not*.

18

Note that the distinction between document and text nodes is not required for the basic model and we often ignore text nodes for clarity. Text nodes are required if we support sharing of text by documents and to support the extended model of section 5 which includes citation links and document clustering. If we allow document nodes to share text nodes, then an *or* matrix is appropriate, t_i is true when any parent is instantiated.

Representation concept nodes. Link matrix forms for representation concepts were discussed in section 3.4. For binary indexing and unweighted terms an *or*-combination can be used. For *tf*, *idf*, or *tf.idf* weights a weighted-sum link matrix is used.

Query concept nodes. As we have seen, previous indexing research can be incorporated directly in the document network. The query network, particularly the links between representation and query concepts is less well understood. Here we are interested in estimating the probabilistic dependence of concepts mentioned in the user's query upon the representation concepts. Most current retrieval models view these two sets of concepts as identical under the assumption that the user knows the set of representation concepts and can formulate queries using the representation concepts directly. Under this assumption, the same link matrix as for text nodes should be used.

Research suggests, however, that the mismatch between query and indexing vocabularies may be a major cause of poor recall [FLGD87]. While our initial implementation is limited to linking query concepts to "nearly" equivalent representation concepts using a weighted-sum combination rule, it would appear that improved estimates of the dependence of query concepts on representation concepts could markedly improve performance. Two areas of research bear directly on improving the quality of these estimates: automatic thesaurus construction and natural language research aimed at extracting concept descriptions from query text, identifying synonymous or related descriptions, and resolving ambiguity.

Query nodes. The dependence of query nodes on the query concepts is more straightforward. For Boolean queries we use the procedure described in section 4.3. For probabilistic queries we use a weighted-sum matrix. In both cases we can adjust link matrix values if we have information about the relative importance of the query concepts.

Information need. The information need can generally be expressed as a small number of queries of different types (Boolean, m-of-n, probabilistic, natural language, ...). These can be combined using a weighted-sum link matrix with weights adjusted to reflect any user judgments about the importance or completeness of the individual queries.

5   Extensions to the basic model

The basic model described in section 3 is limited in at least two respects. First, we have assumed that evidence about a variable establishes its value with certainty. Second, we have represented only a limited number of dependencies between variables. In this section we will see that these limitations can be removed.

5.1   Uncertain evidence

The only use of evidence in the basic model is to assert that a document has been observed ($d_i = true$). During query processing we assert each document true and rank documents
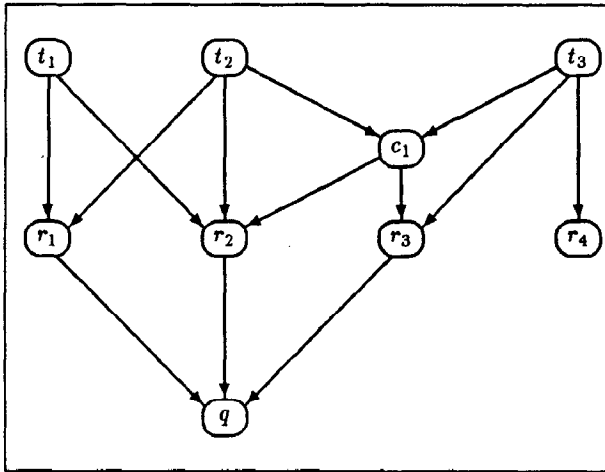
Figure 6: Document clustering model

based on the probability that the information need is met. Evidence is attached to a node a in a Bayesian network by creating a new *evidence* node b as a child of a. This new node b then passes a likelihood vector (both components of a likelihood ratio) to a. Since evidence is expressed in terms of likelihood we are not restricted to the values *true* and *false* but need only specify the likelihood of $a = true$ and $a = false$ given the evidence summarized at b. Thus we can "partially" instantiate nodes in the network when the evidence we have is not sufficient to establish the value of a proposition with certainty. This uncertain evidence can be used to model citation and document cluster information.

Document clustering. A variety of document clustering techniques have been developed for information retrieval [vR79]. Document clustering is generally used to find documents that are similar to a document that is believed relevant under the assumption that similar documents are related to the same queries. Our use of cluster information is somewhat different since we do not retrieve clusters, but we can incorporate cluster information by treating cluster membership as an additional source of evidence about document content. In the fragment shown in Figure 6, document texts t_1, t_2, and t_3 are indexed using representation concepts r_1, r_2, r_3, and r_4. Documents t_2 and t_3 have been identified as part of cluster c_1; both texts are linked to a cluster node and the cluster node is linked to the representation concepts that define the cluster. The cluster node is similar to a conventional cluster representative. Documents t_1 and t_2 are indexed by the same representation concepts (r_1 and r_2) and, if we assume equivalent conditional probabilities, would be ranked equivalently in the absence of the cluster node. With the addition of the cluster node, however, a new representation concept (r_3) is associated with t_2 by virtue of its cluster membership. Assuming that r_3 contributes positively to the belief in q, t_2 would be ranked higher than t_1. In practice, the links between documents and clusters are not represented in the network; evidence is attached to all clusters to which a document has been assigned when that document is instantiated.

Citation and nearest neighbor links. A variety of asymmetric relationships between pairs of documents can also be represented. These relationships are similar to clustering in that they use an assumed similarity between documents to expand the set of representation concepts that can be plausibly associated with a text. They differ in that they are ordered relations defined on pairs of documents rather than an unordered, set membership relationship between documents and clusters.

One example of this kind of relationship is the nearest neighbor link in which a document is linked to those documents judged to be most similar to the original. A second example is based on citations occurring in the text. Citation links may be useful if the type of reference can be determined to allow estimation of the probabilistic dependence between the nodes. Again, these links are not explicitly represented in the network; evidence is attached to a document's nearest neighbors and citation partners when the document is instantiated.

5.2 Additional dependencies

In the basic model, we assume that there are no dependencies between documents, between texts, between representation concepts, between query concepts, or between queries. While independence assumptions like these are common in retrieval models, it is widely recognized that the assumptions are unrealistic. In addition to the document cluster and citation information which is modeled as evidence, we would like to explicitly represent the term dependencies embodied in conventional term clusters and thesauri.

The basic mechanism for representing these dependencies is unchanged, we identify the set of nodes upon which a given node depends and characterize the probability associated with each node conditioned on its immediate parents. When adding these new links, however, we must be careful to preserve the acyclic nature of the inference network. Bayesian inference networks cannot represent cyclic dependencies, in effect evidence attached to any node in the cycle would continually propagate through the network and repeatedly reinforce the original node. In the basic model, no cycles are possible since nodes are only linked to node types that are lower in the DAG. The introduction of these new dependencies makes cycles possible.

Inference networks provide a natural mechanism for representing dependencies between representation concepts and between query concepts. Several automatic clustering techniques produce structures that can be used in an inference network. For example, dependence trees or Chow trees [vR79,Pea88] contain exactly the term dependence information required for an inference network in a form that is guaranteed to be cycle-free.

These networks can also be used to represent probabilistic thesaurus relationships. These relationships extend those of a conventional thesaurus by including conditional probability information. For example, a conventional thesaurus might list "house pet" as a broader term for "dog" and "cat"; the network representation will include a specification of the probability that "house pet" should be assigned given a document containing "dog" or "cat" in isolation, neither term, or both terms.

Synonyms, related terms, and broader terms can be represented by creating a new node to represent the synonym or related term class or the broader term and adding the new node as a child to the relevant representation concept nodes. We will generally prefer to

add these nodes as part of the query network since their presence in the document network would represent a computational burden even when not used in a query. Although generally less useful, narrower term relationships can also be represented.

6   Conclusion

The retrieval model presented here provides a framework within which to integrate several document representations and search strategies. We are currently refining the model and conducting experiments to compare search performance based on this model with that of other models and to compare performance of potential representations and search strategies.

References

[CH79]   W. Bruce Croft and D. J. Harper. Using probabilistic models of document retrieval without relevance information. *Journal of Documentation*, 35:285–295, 1979.

[Che88]   Peter Cheeseman. An inquiry into computer understanding. *Computational Intelligence*, 4:58–66, February 1988. Article is part of a debate between logic and probability schools in AI.

[CK87]   Paul R. Cohen and Rick Kjeldsen. Information retrieval by constrained spreading activation in semantic networks. *Information Processing and Management*, 23(2):255–268, 1987.

[Coh85]   Paul R. Cohen. *Heuristic Reasoning About Uncertainty: An Artificial Intelligence Approach*. Pitman, Boston, MA, 1985.

[Coo71]   W. S. Cooper. A definition of relevance for information retrieval. *Information Storage and Retrieval*, 7:19–37, 1971.

[Cro80]   W. Bruce Croft. A model of cluster searching based on classification. *Information Systems*, 5:189–195, 1980.

[Cro86]   W. Bruce Croft. Boolean queries and term dependencies in probabilistic retrieval models. *Journal of the American Society for Information Science*, 37(2):71–77, 1986.

[Cro87]   W. Bruce Croft. Approaches to intelligent information retrieval. *Information Processing and Management*, 23(4):249–254, 1987.

[CT87]   W. Bruce Croft and Roger H. Thompson. I³R: A new approach to the design of document retrieval systems. *Journal of the American Society for Information Science*, 38(6):389–404, November 1987.

[CT89]   W. Bruce Croft and Howard Turtle. A retrieval model incorporating hypertext links. In *Hypertext '89 Proceedings*, pages 213–224, 1989.

[Dem68] A. P. Dempster. A generalization of Bayesian inference. *Journal of the Royal Statistical Society B*, 30:205–247, 1968.

[Doy79] John Doyle. A truth maintenance system. *Artificial Intelligence*, 12(3):231–272, 1979.

[FLGD87] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, November 1987.

[FNL88] Edward A. Fox, Gary L. Nunn, and Whay C. Lee. Coefficients for combining concept classes in a collection. In *Proceedings of the Eleventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 291–308, New York, NY, 1988. ACM.

[Fuh89] Norbert Fuhr. Models for retrieval with probabilistic indexing. *Information Processing and Management*, 25(1):55–72, 1989.

[KL86]   Laveen N. Kanal and John F. Lemmer, editors. *Uncertainty in Artificial Intelligence*. North-Holland, Amsterdam, 1986.

[KMT⁺82] J. Katzer, M. J. McGill, J. A. Tessier, W. Frakes, and P. DasGupta. A study of the overlap among document representations. *Information Technology: Research and Development*, 1:261–274, 1982.

[LK88]   John F. Lemmer and Laveen N. Kanal, editors. *Uncertainty in Artificial Intelligence 2*. North-Holland, Amsterdam, 1988.

[LS88]   S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society B*, 50(2):157–224, 1988.

[MK60]   M. E. Maron and J. L. Kuhns. On relevance, probabilistic indexing and information retrieval. *Journal of the ACM*, 7:216–244, 1960.

[MKN79] Michael McGill, Mathew Koll, and Terry Noreault. An evaluation of factors affecting document ranking by information retrieval systems. Technical report, Syracuse University, School of Information Studies, 1979. Funded under NSF-IST-78-10454.

[Nil86] Nils J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28(1):71–87, 1986.

[OPC86]   Robert N. Oddy, Ruth A. Palmquist, and Margaret A. Crawford. Representation of anomalous states of knowledge in information retrieval. In *Proceedings of the 1986 ASIS Annual Conference*, pages 248–254, 1986.

[Pea88]   Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

[Rob77]   S. E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, December 1977.

[Sal88]   Gerard Salton. A simple blueprint for automatic boolean query processing. *Information Processing and Management*, 24(3):269–280, 1988.

[Sha76]   Glen Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[SM83]   Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[Sti75]   K. H. Stirling. The effect of document ranking on retrieval system performance: A search for an optimal ranking rule. *Proceedings of the American Society for Information Science*, 12:105–106, 1975.

[TC89]   Roger H. Thompson and W. Bruce Croft. Support for browsing in an intelligent text retrieval system. *International Journal of Man-Machine Studies*, 30:639–668, 1989.

[TS85]   Richard M. Tong and Daniel Shapiro. Experimental investigations of uncertainty in a rule-based system for information retrieval. *International Journal of Man-Machine Studies*, 22:265–282, 1985.

[vR79]   C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

[vR86]   C. J. van Rijsbergen. A non-classical logic for information retrieval. *Computer Journal*, 29(6):481–485, 1986.

[Wil73]   Patrick Wilson. Situational relevance. *Information Storage and Retrieval*, 9:457–471, 1973.

[Zad83]   Lotfi A. Zadeh. The role of fuzzy logic in the management of uncertainty in expert systems. *Fuzzy Sets and Systems*, 11:199–228, 1983.