Evaluation of an Inference Network-Based Retrieval Model

HOWARD TURTLE and W. BRUCE CROFT
University of Massachusetts

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. The performance of a retrieval system based on the inference network model is evaluated and compared to performance with conventional retrieval models.

Categories and Subject Descriptors: H.30 [**Information Storage and Retrieval**]: General; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval— *retrieval models*; H.3.m [**Information Storage and Retrieval**]: Miscellaneous.

General Terms: Experimentation, Performance, Theory

Additional Key Words and Phrases: Document retrieval, inference networks, network retrival 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 [38], document clustering [7], spreading activation search [4], support for multiple search strategies [11], and representation of user knowledge [27] or document content [40].

Recent work suggests that significant improvements in retrieval performance will require techniques that, in some sense "understand" the content of documents and queries [9, 43] 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 [4, 12].

This work was supported in part by the Air Force Office of Scientific Research under contract 90-0110 and by NSF grant IRI-8814790.
Authors' Address: Computer and Information Science Department, University of Massachusetts, Amherst, MA 01003. email: croft@cs.umass.edu

The idea that retrieval is an inference or evidential reasoning process is not new. Cooper's logical relevance [6] is based on deductive relationships between representations of documents and information needs. Wilson's situational relevance [44] 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 [19, 21] 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 do the following:

—Support the use of multiple document representation schemes. Research has shown that a given query will retrieve different documents when applied to different representations, 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 [20]. Similar results have been obtained when comparing term- with cluster-based representations [2] and term- with citation-based representations [16].

—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 [20, 24]. 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 [9].

—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 [15]. 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 (Section 2), present an inference network-based retrieval model (Sections 3 and 5), compare the network model to current retrieval models (Section 4), and evaluate the performance of an implementation of the network model (Sections 6 and 7).

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 [19, 21]. Popular approaches include those based on purely symbolic reasoning [5, 14], fuzzy sets [45], and a variety of probability models [3, 25]. Two inference models based on probabilistic methods are of particular interest: Bayesian inference networks [22, 28] and the Dempster-Shafer theory of evidence [13, 34].

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 $P(q \mid p)$ 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 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 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.

[1]While this probability specification is generally referred to as a link matrix, it is actually a tensor.

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 $P(b \mid a) = p$ and $P(b \mid \neg a) = 0$, the Bayesian and Dempster–Shafer models will produce similar results [28, 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 [23]. 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.

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 the query is refined or additional representations 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).

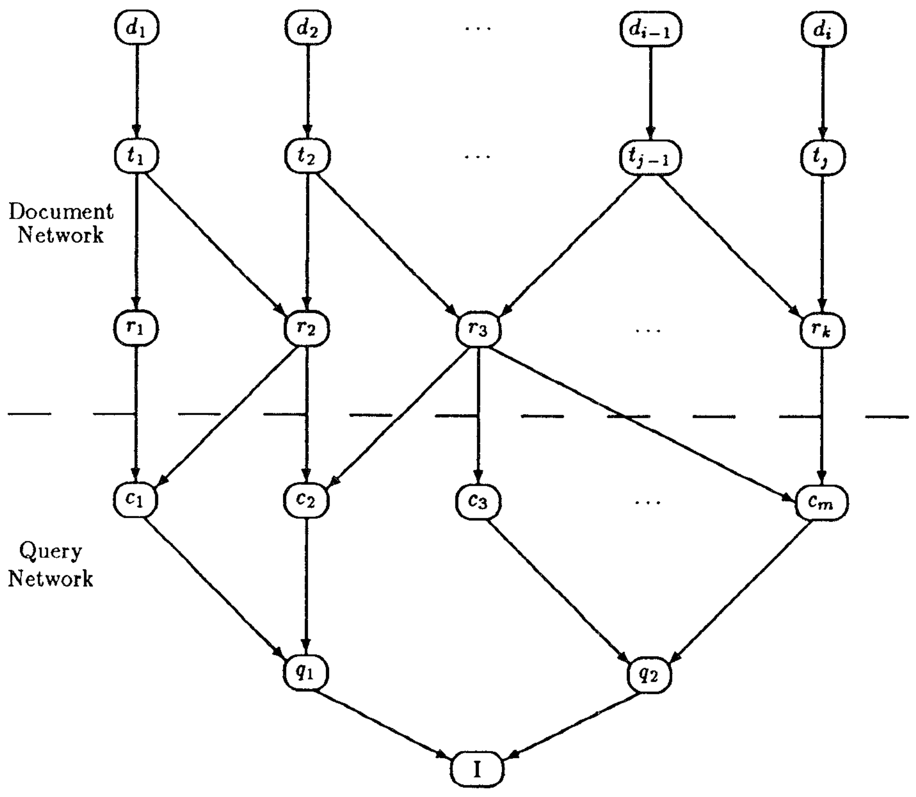

Fig. 1. Basic document inference network.

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 document 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 are rare in traditional collections (an example would be a journal article that appears in both a serial issue and in a reprint collection) and are not generally represented in current retrieval models, they are 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 weight (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 [40].

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 our belief that a user's information need is met on the documents in a collection where the dependence in 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 information retrieval [37], and similar problems arise in pattern recognition, medical diagnosis, and truth-maintenance systems. For a discussion of the best-subset or belief revision problem in Bayesian networks see Pearl [28]. 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 [29] 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 [36].

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 $P(a \mid p_1, \ldots, p_n)$.

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 p rovide diagnostic information to the set of parents based on our belief in a. As suggested earlier, 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

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

For *or*-combinations, Q will be true when any 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

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

In this representation the first row corresponds to the case that $Q = false$ and the second row corresponds to $Q = true$. Each column corresponds to a specific combination of parent values. We number the columns from 0 to 7 and use the binary representation of the column number to index values of A, B, and C. Thus, column 0 (000_2) corresponds to the case in which $A = false$, $B = false$, and $C = false$. Column 1 (001_2) corresponds to the case in which $A = false$, $B = false$, and $C = true$. Similarly, column 7 (111_2) corresponds to the case in which all parents are true.

To compute our belief in $Q = true$, we simply compute a probability for each possible combination of parent variables and multiply that probability by the appropriate matrix element (second row). Since column 0 has a value of 0.0 and all remaining columns have a value of 1.0, this gives

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

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*

$$P(Q = true) = abc \tag{2}$$

and *not*

$$P(Q = true) = 1 - a. \tag{3}$$

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

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

Evaluation of this link matrix form results in

$$P(Q = true) = \frac{(w_a a + w_b b + w_c c) w_q}{t}. \tag{4}$$

This link matrix can be used to implement a variety of weighting schemes, including the familiar term weighting schemes based on the frequency of a term in a single document (*tf*), and its frequency in the entire collection (*idf*) or both (*tf.idf*). These *tf.idf* weights are dicussed in more detail in Section 7.1.

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 normalized *tf* values for A, B, and C, let idf_q be a normalized *idf* weight for Q, and let

$$w_q = idf_q \cdot (w_a + w_b + w_c). \tag{5}$$

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

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

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 nonrelevant 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 5 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 inference networks used in experiments 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.

4.1 Probabilistic Retrieval Models

Conventional probabilistic models [36, 42] rank documents by the probability that each document would be judged relevant to a given query, $P(\text{relevant} \mid d_i)$.[2] This is, in many ways, similar to computing the probability that a user's information need is met given a specific document, $P(I \mid d_i)$. 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 $P(I \mid d_i)$ (or $P(Q \mid d_i)$ 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 [42] is shown in Figure 2. A document is represented by a vector whose components are indexing or representation concepts ($d_i = \{r_1, \dots, r_n\}$). 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

[2]Most probabilistic models do not actually compute $P(\text{relevant} \mid d_i)$, but simply rank documents using some function that is monotonic with $P(\text{relevant} \mid d_i)$. Like Fuhr [18], 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.

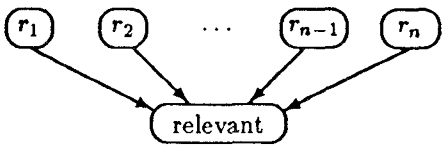

Fig. 2. Inference network for binary independence model.

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 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 nonrelevant 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 nonrelevant documents, it is easy to compute $P(r_i)$ for the relevant sample and to use the result as an estimate of $P(r_i \mid \text{relevant} = \textit{true})$. We can use a similar estimate for $P(r_i \mid \text{relevant} = \textit{false})$. Given a set of independence assumptions and estimates for $P(d_i)$ and $P(\text{relevant} = \textit{true})$ we can compute $P(\text{relevant} \mid d_i)$.[3] Estimating $P(\text{relevant} \mid d_i)$ without the use of Bayes' rule would be extremely difficult.

Essentially the same procedures can be used to estimate $P(Q \mid d_i)$. The main difference between the two estimates is that instead of using the representation concepts directly we must compute $P(c_j \mid \pi_{c_j})$ for each query concept and use these beliefs to estimate $P(Q \mid d_i)$.

The question remains, however, whether estimates of $P(\text{relevant} \mid d_i)$ or $P(Q \mid d_i)$ obtained in this way match users' intuition about the dependence. The fact that relevance feedback does improve retrieval performance suggests that the estimates of $P(\text{relevant} \mid d_i)$ 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 $P(\text{relevant} \mid d_i)$ directly from a small number of documents is impractical, it may be possible to obtain estimates of $P(Q \mid \pi_Q)$. Users may, for example, be able to assign importance to the concepts in their query and may be able to identify significant interactions

[3] $P(d_i)$ and $P(\text{relevant} = \textit{true})$ do not play a major role in probabilistic models that only produce a document ranking but are required to compute $P(\text{relevant} \mid d_i)$.

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 $P(\text{relevant} \mid d_i)$. The model presented here views the problem of estimating $P(I \mid d_i)$ as consisting of a set of logically related estimates. Each estimate is made independently using procedures specific to the type of estimate; the "probabilistic" estimate of $P(Q \mid \pi_Q)$ 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 $P(I \mid d_i)$.

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) [18]. 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, $P(x_i = 1 \mid d_m)$ 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 $\mathbf{x}$, where $|X| \leq 2^n$.

The network shown in Figure 3 corresponds to the probability distribution

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

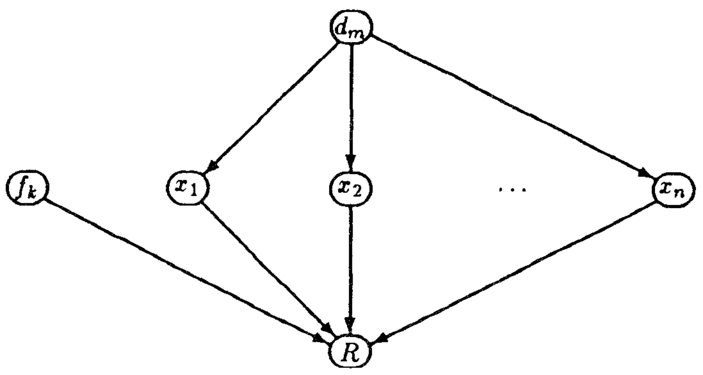

Fig. 3. Inference network for RPI model

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

$$P(R \mid f_k, d_m) = P(R \mid f_k, x_1, \dots, x_n) P(x_1 \mid d_m) \dots P(x_n \mid d_m).$$

Assuming that the descriptors are assigned independently, that is

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

the basic ranking expression for the network of Figure 3 is

$$P(R \mid f_k, d_m) = \sum_{x \in X} P(R \mid f_k, \mathbf{x}) P(\mathbf{x} \mid d_m). \tag{6}$$

Equation 6 is equivalent to the basic ranking expression used by Fuhr [18, equation 9]. Equation 6 can be expanded to the product form

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

where

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

(Strictly speaking the network corresponding to (6) should have a single node $\mathbf{x}$ in place of $x_1, \dots, x_n$ since (6) makes no independence assumptions. Independence is, however, assumed in all derivations based on (6) 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

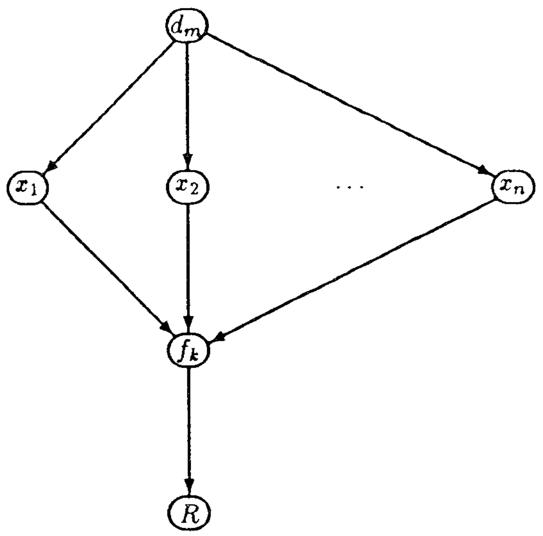

Fig. 4. Example inference network.

the probability distribution

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

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

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

Applying Bayes' rule we get

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

Assuming that the x_i are distributed independently in documents (8) and that the assignment of the x_i is independent of the query (9)

$$P(x_1, \ldots, x_n) = \prod_{1 \le i \le n} P(x_i) \tag{8}$$

$$P(x_1, \ldots, x_n \mid f_k) = \prod_{1 \le i \le n} P(x_i \mid f_k) \tag{9}$$

we have

$$P(R \mid d_m) = P(R \mid f_k) P(f_k) \sum_{\mathbf{x} \in \mathbf{X}} \prod_{1 \le i \le n} \frac{P(x_i \mid f_k)}{P(x_i)} P(x_i \mid d_m). \tag{10}$$

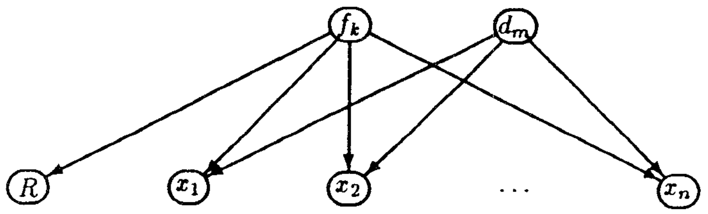

Fig. 5. Effect of inversion.

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 $P(f_k \mid x_1, \ldots, x_n)$ and would not invert the network.

Equation 10 reduces to

$$
\begin{aligned}
&P(R \mid d_m) \\
&\quad = P(R \mid f_k)P(f_k) \prod_{1 \le i \le n} \left(\frac{P(x_i = 1 \mid f_k)}{P(x_i = 1)} P(x_i = 1 \mid d_m) \right. \\
&\quad \left. + \frac{P(x_i = 0 \mid f_k)}{P(x_i = 0)} P(x_i = 0 \mid d_m) \right).
\end{aligned}
$$

If we let

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

we get the ranking expression

$$
P(R \mid d_m) = P(R \mid f_k)P(f_k) \prod_{1 \le i \le n} \left(\frac{p_{ik}}{q_i} u_{im} + \frac{1 - p_{ik}}{1 - q_i} (1 - u_{im}) \right). \quad (11)
$$

Equation (11) differs from (7) 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

[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 $P(x_i \mid f_k, d_m)$ 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 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.

x cannot influence our assessment of relevance except through its effect on the query. The network of Figure 3 assumes that **x** and f_k are independent, but not necessarily conditionally independent given R, that is, **x** and the query can influence our assessment of relevance independently. Under the assumption of conditional independence

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

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 as a separate variable that is related to the possible document descriptions through the specification of $P(R \mid \mathbf{x}, f_k)$. 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 $\text{bel}(r_i) = 1$. All remaining representation concepts have $\text{bel}(r_j) = 0$.

(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 $\text{bel}(Q) = 1$ satisfies the query, any node for which $\text{bel}(Q) < 1$ does not.

Under the assumptions above and using binary indexing, $\text{bel}(Q)$ 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 $\text{bel}(Q) = 1$ satisfy the query and all remaining documents have a small but nonzero $\text{bel}(Q)$.

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, (1) through (4) provide a natural interpretation of Boolean operations in probabilistic terms and can be used with binary or weighted indexing.

The binary nature of the retrieval decision in Boolean systems is frequently cited as a drawback [8, 30, 36]. 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 $n \le c \le \infty$ and interpret the *and* operator to mean

$$P(Q_{and} = true \mid n \text{ parents} = true) = 1$$

$$P(Q_{and} = true \mid k \text{ parents} = true) = 1 - \frac{n-k}{c}, \qquad 0 > k > n$$

$$P(Q_{and} = true \mid no \text{ parents} = true) = 0$$

and the *or* operator to mean

$$P(Q_{or} = true \mid n \text{ parents} = true) = 1$$

$$P(Q_{or} = true \mid k \text{ parents} = true) = \frac{k}{c}, \qquad 0 < k < n$$

$$P(Q_{or} = true \mid no \text{ parents} = true) = 0.$$

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 [30, 36]. 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*.

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 the 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 [15]. 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 probabilisitic 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 as true and rank documents 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 [42]. 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

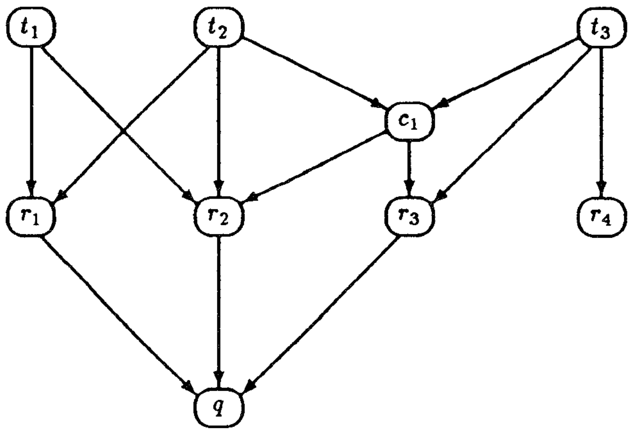

Fig. 6. Document clustering model.

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 the document is instantiated.

Citation, nearest neighbor, and hypertext links. A variety of asymmetric relationships between pairs of documents can also be represented. Croft and Turtle [12] discuss the use of inference networks to model hypertext links. Citation and nearest neighbor relationships are similar to clustering in that an assumed similarity between documents can be used to expand the set of representation concepts that can be plausibly associated with a text. They differ from clustering in that they are ordered relations defined on pairs of documents rather than an unordered, set membership relationship between documents and clusters.

Once 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 phrases, conventional term clusters [42], 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 since, 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 [28, 42] 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. EVALUATION METHODOLOGY

To test the effectiveness of the inference network retrieval model, inference networks were built for two test collections. The CACM test collection contains 3204 records describing articles published in the *Communications of the ACM* from 1958 to 1979. CACM records contain author, title, abstract, and citation information as well as manually assigned keywords and Computing Review categories. The CISI test collection contains 1460 records describing highly cited information science articles published between 1969 and 1977. CISI records contain author, title, abstract, and citation information. Queries and relevance judgments are provided with each collection. A

Table I. Selected Collection Statistics

	CACM	CISI
collection size	3204	1460
unique stems	5493	5448
maximum stem frequency	1333 (algorithm)	660 (inform)
stem occurrences	117578	98304
postings	79243	71017
max within document frequency	27	27
mean within document frequency	3.7	5.2
queries	50	35

query in these collections is a natural language description of an information need. Boolean queries have been manually constructed from these natural language descriptions by the test collection providers. One set of Boolean queries is available for CISI and two sets (referred to as BL1 and BL2) are available for CACM. Summary statistics for both collections are shown in Table I. For a more detailed description of the test collections and their history, see Fox [17].

Most of the experimental work was conducted with the CACM collection since it is widely used and allows comparison of our results with a large body of previous work. Experiments with the CISI collection were carried out to validate results obtained with the CACM collection since the performance of many retrieval techniques is collection dependent. Previous studies have shown that absolute performance on the CISI collection is low when compared to most other test collections, including CACM. CISI articles tend to be general and queries tend to be vague (e.g., "What is information science?").

Retrieval performance is described in terms of precision and recall. Precision/recall data will be presented in tabular form showing precision at ten standard recall points. When two tests are being compared, we show the difference as the percent change from the baseline test. Significance tests are based on a one-tailed Sign test [35] comparing the ten averaged precision values for each query set, where a 5 percent difference in average precision is required for two observations to be considered different. This is a conservative test, but it makes few assumptions about the distribution of observed data. As an informal rule, a difference of 5 percent in average precision is generally considered significant, and a 10 percent difference is considered very significant [31].

7. RESULTS

The research hypotheses discussed here are:

(1) Given equivalent document representations, query forms, and assumptions about the match between indexing and query vocabularies, the inference network model will perform as well as conventional probabilistic models.

(2) The use of multiple query formulations and search strategies will significantly improve retrieval performance when compared to equivalent networks with a single natural language query.

Computational aspects of the model are discussed in [39] and complete results are presented in [41].

In the remainder of this section, we discuss techniques for estimating the required probabilities (Section 7.1), compare performance with simple probabilistic searches (Section 7.2), and give results for multiple query representations (Section 7.3).

7.1 Estimating the Probabilities

For the basic model we must provide two probability estimates: one estimate characterizes the dependence of a query or information need upon the terms in the collection (Section 7.1.1) and the second characterizes the dependence of individual terms on their parent documents (Section 7.1.2).

7.1.1 *Dependence of Queries on the Document Network.* As suggested in Section 3.4, the procedures for estimating $P(Q \mid t_i)$ differ for Boolean and natural language queries. Estimates for Boolean queries use product-form estimates which are largely determined by the model, while estimates for natural language queries use weighted sums and are based on the results of previous information retrieval research.

Natural language queries. A natural language query is simply a natural language description of an information need. Words in the query that generally do not affect retrieval performance (stop words) are removed, and the remaining words are stemmed to remove common endings in an attempt to reduce simple spelling variations to a single form. Documents are then ranked using Eq. 4 to combine probabilities.

Two factors are commonly used in weighting the contribution of query terms—the frequency of the term in the query (*qf*) and the inverse document frequency (*idf*) of the term in the collection. The basic ideas are that (1) a content-bearing term that occurs frequently in the query is more likely to be important than one that occurs infrequently, and (2) those terms that occur infrequently in the collection are more likely to be important than frequent or common terms.

The use of query term weights based on *idf* alone decreased retrieval performance. Weights based on within-query frequency (*qf*) and the combination of *qf* and *idf* both increased retrieval performance (significance levels of 0.002 for both weightings). In general, the performance levels of *qf* and *qf.idf* weightings are quite similar, and the choice of one over the other will depend on the function used for $P(t_i \mid d_j)$ and on the query types to be used. For many applications, the *qf* weighting will be preferred since it is simpler and performs at least as well as *qf.idf* at high precision. All results reported here use either *qf* or *qf.idf* weighting.

Boolean queries. A Boolean query consists of an expression using the operators *and*, *or*, and *not* with query terms as operands. Stopwords are removed from the queries, query terms are stemmed as with probabilsitic queries, and documents are ranked using Eqs. 1 through 3 to combine probabilities.

For conventional Boolean queries, then, the retrieval model specifies how probabilities are to be combined and no additional probabilities need be estimated. Since it would be straightforward to add weights to terms in Boolean queries, an open research issue is whether significant improvements can be obtained by weighting Boolean query terms. While the Boolean queries available with the standard test collections do not include term importance information, [1] suggests that useful information about term importance can be readily obtained from users; this information could be directly incorporated as weights for Boolean expression evaluation.

It is also possible that some form of *idf* weighting could be used to improve performance of Boolean queries or that weighting strategies developed for conventional Boolean systems [26] could be adapted to the network model. An experiment in which each intermediate *and* and *or* expression was weighted using an *idf* based on the number of documents in the intermediate result did not improve performance.

7.1.2 *Dependence of Term Belief on Documents.* The probability that a term accurately describes the content of a document can be estimated in several ways, but previous information retrieval research has consistently shown within-document frequency (*tf*) and inverse document frequency (*idf*) to be useful components of such estimates [32]. In developing estimates, we concentrated on functions involving *tf* and *idf*; other functions are certainly possible and could be used in the basic model. The *idf* measure used in these experiments is given by

$$idf = \frac{\log\left(\frac{\text{collection size}}{\text{term frequency}}\right)}{\log(\text{collection size})}.$$

The measure used for within-document frequency in these experiments is a normalized *tf* [36, 42] in which the score for term i in document j is given by

$$ntf_{ij} = \frac{\text{frequency of term } i \text{ in doc } j}{\text{max frequency for any term in doc } j}.$$

Given these basic forms for the *tf* and *idf* estimates, several experiments were conducted to determine the following:

(1) What range of belief values for a term is appropriate given that a term occurs in an instantiated document ($P(t_i \mid d_j = \text{true})$).
(2) What range of belief values is appropriate given that a term does not occur in an instantiated document ($P(t_i \mid d_j = \text{false})$).
(3) How the *tf* and *idf* components should be combined when forming the overall estimate of term belief.

Strictly speaking, our use of $P(t_i \mid d_j = \text{true})$ and $P(t_i \mid d_j = \text{false})$ here is a bit loose. In the basic model, only one parent document is instantiated at a time, so that any term has either exactly one instantiated parent or no instantiated parents. When we write $P(t_i \mid d_j = \text{true})$, we really mean

$$P(t_i \mid d_j = \text{true} \wedge \text{all other parents} = \text{false}).$$

When we write $P(t_i \mid d_j = \text{false})$, we really mean

$$P(t_i \mid \text{all parents} = \text{false}).$$

We occasionally refer to $P(t_i \mid d_j = \text{true})$ as the belief in t_i, given that parent d_j is true, and will refer to $P(t_i \mid \text{all parents} = \text{false})$ as the default belief for t_i.

Default belief. Most probabilistic models form a document score by adding the scores for each term in common with the query; missing terms are treated as if they had a score of 0. In the inference network model, this amounts to asserting that the absence of a term in a document (in most cases, in the abstract or title of the document) implies certainty that the term should not be assigned to the document. Given the small size of the title and abstract compared to the full document and the fact that the query and document vocabularies are not identical, a less extreme estimate for the probability that a term should be assigned, given that it is not observed, seems more reasonable. This estimate should probably depend on the type of document record in use—the absence of a term from the full text of a journal article should be treated as stronger evidence than its absence from abstract or title.

Since the within-document frequency for all of these term-document pairs is zero, we considered only estimates based on a term's collection frequency (*idf*) and hypothesized that the default belief should be proportional to the frequency of the term in the collection (inversely proportional to *idf*). Essentially, this hypothesis asserts that a common term is more likely to be a correct descriptor than a rare term given that neither occurs in the document.

Retrieval performance for natural language queries with a fixed default value is assigned to all terms (no *idf* weights) peaks for defaults in the range 0.3 to 0.4. A higher default probability reduces the effect of a missing term on a document's ranking, but it also compresses the range of belief values for the query given the documents in the collection.

Since values in the 0.3 to 0.4 range produced the best retrieval performance, several default estimates of the form $\alpha - (\beta * idf)$ were tried, where α ranged from 0.2 to 0.5 and β ranged from 0.2 to 0.4 (estimates for $P(t_i \mid d_j = \text{true})$ were adjusted as necessary to avoid overlapping ranges). The *idf*-weighted defaults perform about as well as the fixed defaults.

Since a default estimate that is inversely proportional to a term's *idf* did not consistently improve performance, estimates in which the default is directly proportional to *idf* were examined. With this interpretation, the absence of a common term is stronger evidence that the term should not be assigned to the document than the absence of a rare term. These estimates performed significantly worse than either the fixed or the original *idf*-weighted estimates.

Combining the tf and idf estimates. A large number of functions for combining the *tf* and *idf* estimates were tested. These functions have the general form

$$P(t_i \mid d_j = \text{true}) = \alpha + \beta^* tf + \gamma^* idf + \delta^* tf^* idf$$

where $0.4 \le \alpha \le 0.6$ and β, γ, and δ were chosen to produce a probability in the range [0..1]. The best performance is generally achieved when $\beta = \gamma = 0.0$ and only the *tf.idf* product term remains. In a final set of experiments, changes to the relative weight of the *tf* and *idf* components of this product term were tested, but changing weights did not significantly improve performance.

Several ranking functions were developed during these experiments that perform well when compared to conventional retrieval models. The performance of many of the best functions is quite similar, but a good overall belief estimate is given by

$$\begin{aligned} P(t_i \mid d_j = \text{true}) &= 0.4 + 0.6^* tf^* idf \\ P(t_i \mid \text{all parents false}) &= 0.4. \end{aligned} \tag{12}$$

Variations that work about as well use a log normalization for the *tf* component or an *idf*-weighted default (e.g., $P(t_i \mid \text{all parents false}) = 0.4 - (0.2^* idf)$).

While this estimate works well for both collections and for all query types, it is probably not the best that can be achieved. Our objective in these experiments was to gain a better understanding of the major factors that influence the performance of the retrieval model and how traditional information retrieval weightings could be implemented in the inference network model. Further research can certainly improve these estimates, but it is difficult to estimate how much additional performance can be gained.

7.2 Baseline Results

Given the strategies for estimating belief at nodes in the network, we now wish to compare the performance of the network model with that of conventional probabilistic (Section 7.2.1) and Boolean (Section 7.2.2) models.

7.2.1 *Probabilistic Retrieval.* It is possible to build inference networks that are equivalent to conventional probabilistic systems if exactly the same indexing strategies are used. In practice, minor variations in the way documents are parsed and indexed will result in small performance differences even when the network form implements a ranking function that is equivalent to that used in the probabilistic system. Using the estimates of the last section, however, it is possible to build networks that perform better than conventional probabilistic models. Table II shows retrieval performance of the network model compared to a baseline probabilistic model that uses *tf.idf* weighting. Performance improves 25.0% for the CACM collection and 5.3%

Table II. Comparison of Probabilistic and Network Model Performance

	Precision (% change) – 50 queries			
	CACM		CISI	
Recall	Probabilistic	Network	Probabilistic	Network
10	60.2	67.2 (+11.7)	34.8	37.3 (+7.0)
20	48.3	56.2 (+16.4)	26.3	29.1 (+10.6)
30	41.0	47.6 (+16.0)	20.4	21.5 (+5.5)
40	30.9	41.3 (+33.6)	17.0	18.4 (+8.3)
50	26.5	34.4 (+30.2)	15.0	15.9 (+5.8)
60	21.6	29.1 (+34.8)	13.2	13.5 (+2.2)
70	15.0	20.3 (+35.5)	10.7	11.6 (+8.3)
80	11.7	16.3 (+39.1)	9.3	9.2 (−1.7)
90	6.4	11.3 (+76.1)	7.4	7.1 (−4.8)
100	4.4	8.7 (+99.8)	5.5	4.8 (−14.0)
average	26.6	33.3 (+25.0)	16.0	16.8 (+5.3)

for CISI. The performance of the network model is better than the probabilistic model at a significance level of 0.001 for CACM and 0.062 for CISI.

These results lead us to accept Hypothesis 1. The demonstrated performance improvements can be attributed primarily to the use of a default probability. Probabilistic models could be formulated to use the same default strategy and to achieve similar improvements.

7.2.2 *Boolean Retrieval.* Establishing a reasonable baseline for Boolean queries is problematic since conventional Boolean retrieval does not rank documents. In order to simulate Boolean queries, we built inference networks with binary belief estimates for $P(t_i \mid d_j)$ and used the normal network evaluation procedures. This effectively assigns all documents that satisfy the query a belief of 1.0, all those that do not satisfy a belief of 0.0, and breaks ties by sorting on document identifier which places newer documents higher in the ranking. Table III compares performance of the network model with conventional Boolean retrieval. Performance improves by 57.7% (BL1) and 49.1% (BL2) for CACM and by 65.3% for CISI. The performance of the network model is better than the Boolean model at a significance level of 0.001 for both CACM query sets and 0.001 for CISI.

Our results are compared with those reported for the Extended Boolean or p-norm model of Salton et al. [33] in Table IV. For this comparison, we use average precision at three recall levels (25%, 50%, and 75%) rather than our customary ten levels, in order to permit comparison with the published results. Note that, for the CACM collection, Salton, et al. [33] include two queries (author searches) that are not used in our experiments.

For both CACM and CISI, the network evaluation of the Boolean queries is better than the best p-norm evaluation. For CACM, average precision for network Booleans is 15.1% better than for the p-norm model for one set of Boolean queries and 7.8% better for the other set. Note that these results should be interpreted cautiously, since the details of the precision/recall computation for the published p-norm results are not known.

Table III. Comparison of Boolean and Network Models

	Precision (% change)				
	CACM			CISI	
Recall	Boolean	BL1	BL2	Boolean	Network
10	46.2	67.6 (+46.3)	66.2 (+43.1)	23.7	45.3 (+91.3)
20	39.3	58.8 (+49.9)	55.5 (+41.5)	20.9	32.8 (+57.2)
30	32.1	50.2 (+56.3)	46.9 (+46.3)	14.6	24.6 (+68.5)
40	26.1	45.2 (+72.8)	41.4 (+58.2)	13.1	20.5 (+56.2)
50	23.6	39.8 (+68.3)	35.4 (+49.8)	12.1	17.9 (+47.5)
60	21.3	33.6 (+57.7)	29.5 (+38.7)	9.2	15.0 (+62.8)
70	13.7	22.0 (+60.0)	22.5 (+63.9)	6.9	12.7 (+85.0)
80	10.5	18.1 (+72.8)	17.8 (+70.3)	5.6	9.5 (+69.0)
90	6.9	11.4 (+65.1)	11.0 (+58.8)	4.9	7.1 (+45.1)
100	5.0	7.9 (+58.9)	8.8 (+75.8)	3.8	4.3 (+14.2)
average	22.5	35.4 (+57.7)	33.5 (+49.1)	11.5	19.0 (+65.3)

Table IV. Average Precision for p-Norm and Inference Network Booleans

	Average Precision – 3 recall points	
	Best p-norm	Network Boolean
CACM	33.1 (p=2)	38.1 (+15.1%)–BL1
		35.6 (+7.8%)–BL2
CISI	18.4 (p=1)	19.2 (+4.3%)

Salton et al. [33] report that the best p-norm interpretation of the Boolean queries outperforms cosine-correlation searches for both CACM and CISI. Our results also show that the Boolean interpretation consistently outperforms the probabilistic (see Table V). This result clearly depends on the quality of the Boolean queries that were generated from the original natural language versions. It would be easy to build Boolean queries that perform poorly, and one would expect performance to improve if domain knowledge was used to expand the set of query terms. The Boolean queries in the test set, however, do not add new terms or domain knowledge. Performance improvements appear to arise because the Boolean queries capture structural information in the queries (phrase structure, compound nominals, and negation) that is not exploited in probabilistic queries. The potential for encoding linguistic structure in Boolean or Boolean-like expressions is an important area for future research.

7.3 Multiple Query Representations

In order to test Hypothesis 2:

> The use of multiple query formulations and search strategies will significantly improve retrieval performance when compared to baseline probabilistic searches.

Table V. Comparison of Probabilistic and Boolean Searches

	Precision (% change)				
	CACM			CISI	
Recall	prob.	BL1	BL2	prob.	Boolean
10	67.2	67.6 (+0.6)	66.2 (−1.6)	38.3	45.3 (+18.3)
20	56.2	58.8 (+4.7)	55.5 (−1.2)	27.9	32.8 (+17.3)
30	47.6	50.2 (+5.5)	46.9 (−1.3)	20.0	24.6 (+23.3)
40	41.3	45.2 (+9.4)	41.4 (+0.2)	17.5	20.5 (+17.4)
50	34.4	39.8 (+15.5)	35.4 (+2.8)	15.5	17.9 (+15.3)
60	29.1	33.6 (+15.2)	29.5 (+1.4)	12.9	15.0 (+16.1)
70	20.3	22.0 (+8.1)	22.5 (+10.8)	11.2	12.7 (+13.7)
80	16.3	18.1 (+10.7)	17.8 (+9.0)	9.0	9.5 (+4.7)
90	11.3	11.4 (+0.6)	11.0 (−3.3)	7.0	7.1 (+1.7)
100	8.7	7.9 (−9.4)	8.8 (+0.2)	4.7	4.3 (−8.3)
average	33.3	35.4 (+6.6)	33.5 (+0.7)	16.4	19.0 (+15.7)

Table VI. Performance of Combined Queries on CACM

	Precision (% change from probabilistic)				
Recall	Probabilistic	BL1	Combined	BL2	Combined
10	67.2	67.6	76.2 (+13.3)	66.2	71.9 (+6.9)
20	56.2	58.8	66.3 (+17.9)	55.5	60.0 (+6.8)
30	47.6	50.2	56.3 (+18.3)	46.9	51.8 (+8.9)
40	41.3	45.2	50.9 (+23.4)	41.4	44.3 (+7.3)
50	34.4	39.8	45.4 (+31.6)	35.4	38.7 (+12.4)
60	29.1	33.6	38.8 (+33.1)	29.5	32.4 (+11.1)
70	20.3	22.0	25.1 (+23.5)	22.5	23.6 (+16.0)
80	16.3	18.1	20.4 (+24.8)	17.8	18.9 (+16.0)
90	11.3	11.4	12.4 (+9.1)	11.0	11.7 (+3.7)
100	8.7	7.9	9.2 (+4.8)	8.8	9.2 (+5.5)
average	33.3	35.4	40.1 (+20.5)	33.5	36.3 (+9.0)

Inference networks were built to evaluate both probabilistic and Boolean versions of the query and to combine the results using a weighted-sum link matrix. Table VI shows the effect of combining probabilistic and Boolean queries with CACM; Table VII shows performance for CISI. Combining queries increased performance by 20.5% (BL1) and 9.0% (BL2) for the CACM collection and by 17.8% for CISI. The combined performance is better than for probabilistic queries at significance levels of 0.002 for both CACM query sets and 0.004 for CISI. The combined performance is better than for Boolean queries at significance levels of 0.001 and 0.002 for CACM, but only 0.250 for CISI. These results lead us to accept Hypothesis 2. The performance improvement due to combining queries is one of the most consistent improvements observed in this research; for any reasonable document network, combining

Table VII. Performance of Combined Queries on CISI

	Precision (% change) – 35 queries		
Recall	Probabilistic	Boolean	Combined
10	38.3	45.3 (+18.3)	44.7 (+16.7)
20	27.9	32.8 (+17.3)	36.1 (+29.2)
30	20.0	24.6 (+23.3)	25.0 (+25.0)
40	17.5	20.5 (+17.4)	20.3 (+16.2)
50	15.5	17.9 (+15.3)	18.0 (+16.2)
60	12.9	15.0 (+16.1)	15.2 (+17.1)
70	11.2	12.7 (+13.7)	12.8 (+14.3)
80	9.0	9.5 (+4.7)	9.5 (+5.3)
90	7.0	7.1 (+1.7)	7.0 (+0.7)
100	4.7	4.3 (−8.3)	4.6 (−2.5)
average	16.4	19.0 (+15.7)	19.3 (+17.8)

probabilistic and Boolean queries gives significant performance improvements.

We originally thought that at least part of the performance improvements arose because the two query types were retrieving different relevant documents, so that the combined set contained more relevant documents than retrieved by the separate queries. This is not, however, the case. The documents retrieved by the Boolean queries are a subset of those retrieved by the corresponding probabilistic query. The individuals preparing the Boolean queries rarely added new terms to the query (for CACM, 4 out of 50 queries in each Boolean set contained new terms), and these new terms retrieved no new relevant documents. It appears that the objective in creating the Boolean queries was to capture the structural information present in the natural language versions, not to produce the best possible Boolean queries. If trained searchers were asked to produce high-recall Boolean queries from the natural language descriptions, they would generally use their knowledge of the subject domain and indexing practice to expand the set of terms to include synonyms and related terms. It is likely that these enhanced searches would retrieve relevant documents not found by the probabilistic query and that these queries would perform better than those provided with the test collection.

The observed performance improvement, then, is due entirely to the fact that the normalized sum of the beliefs produces a better ranking than the rankings produced by the probabilistic or Boolean queries alone.

To determine the degree to which the natural language and Boolean rankings agreed for the CACM test queries, correlation coefficients (Pearson product moment) were computed for the rankings produced by each query. The rankings agreed reasonably well (no negative coefficients), ranging from 0.286 to 1.0 for BL1 and from 0.282 to 1.0 for BL2 with mean rank coefficients of 0.731 and 0.735, respectively (using the belief estimates of Eq. 12). Unfortunately, the correlation coefficient does not appear to be a good predictor of the performance of the combined ranking; identical rankings can both be wrong, and dissimilar rankings can be combined to produce a good result.

Attempts to weight the query types either by scaling beliefs to a similar range or assigning fixed weights to the query types did not improve the performance. It is likely, however, that information about the relative quality of the queries (e.g., user judgements) could be used to weight the contribution of each query and to improve performance.

The actual belief values produced by the two query types are quite variable, and it is difficult to predict whether one ranking will dominate the combined result. Probabilistic beliefs tend to be more uniformly distributed between a minimum that is either fixed (fixed default belief) or determined by the number of query terms (*idf*-weighted default) and a maximum determined by the number of terms and the term weights. Boolean belief values depend heavily on the structure of the Boolean expression. In practice, *and*-structured queries generally rank a relatively small number of documents highly, after which belief values drop off rapidly. *Or*-structured queries tend to produce a more uniform distribution of beliefs.

Croft and Thompson [10] found that different query representations or strategies worked better for some queries than others and that it was difficult to predict which strategy would work best with a given query. In their work, they tried to select a query evaluation strategy based on different query features (query length, sum, and average of *idf* weights), but found that these features did not predict which strategy would work well. One of the strengths of the network model is that it is not necessary to predict which query representation will perform well. Given reasonable query representations, the combined performance is better than that achieved by either representation separately.

8. CONCLUSION

With the CACM collection, the network model gives the performance improvements shown in Table VIII when compared to conventional probabilisitic retrieval. For natural language queries, average precision improves 25.0%, for BL1 it improves 33.3%, and for combined queries it improves by 50.7%. The amount of improvement is, of course, dependent upon the baseline used. Comparing the inference network results with the best reported vector-space results gives improvements ranging from -3.9% to $+42.6\%$ (Table IX). Comparing our results to reported p-norm results (Table X) shows improvement of 8.2% to 30.5% over the best p-norm results. As reported in [41], the use of citations as additional evidence increases network performance by roughly 5–8% over the results shown here.

Based on these results, we can accept the hypotheses discussed in this section:

(1) Given equivalent document representations, query forms, and assumptions about the match between indexing and query vocabularies, the inference network model performs better than conventional probabilistic models.

Table VIII Network Model Performance Improvement

	Precision (% change) – 50 queries			
Recall	probabilistic	NL	BL1	combined
10	60.2	67.2 (+11.7)	67.6 (+12.3)	76.2 (+26.5)
20	48.3	56.2 (+16.4)	58.8 (+21.9)	66.3 (+37.3)
30	41.0	47.6 (+16.0)	50.2 (+22.3)	56.3 (+37.2)
40	30.9	41.3 (+33.6)	45.2 (+46.2)	50.9 (+64.8)
50	26.5	34.4 (+30.2)	39.8 (+50.4)	45.4 (+71.4)
60	21.6	29.1 (+34.8)	33.6 (+55.4)	38.8 (+79.5)
70	15.0	20.3 (+35.5)	22.0 (+46.5)	25.1 (+67.2)
80	11.7	16.3 (+39.1)	18.1 (+53.9)	20.4 (+73.5)
90	6.4	11.3 (+76.1)	11.4 (+77.1)	12.4 (+92.2)
100	4.4	8.7 (+99.8)	7.9 (+81.0)	9.2 (+109.5)
average	26.6	33.3 (+25.0)	35.4 (+33.3)	40.1 (+50.7)

Table IX. Average Precision for Vector-Space and Inference Network Models

	Average Precision – 3 recall points		
	Best vector-space	Network NL	NL+ Boolean
[SM83]	30.3	34.9 (+15.2%)	43.2 (+42.6%)
[Sal88]	36.3	34.9 (-3.9%)	43.2 (+19.0%)

Table X. Average Precision for *p*-Norm and Inference Network Models

	Average Precision – 3 recall points		
	Best *p*-norm	Network Boolean	NL+ Boolean
CACM	33.1 (p=2)	38.1 (+15.1%)-BL1	43.2 (+30.5%)
		35.6 (+7.8%)-BL2	30.6 (+16.7%)
CISI	18.4 (p=1)	19.2 (+4.3%)	19.9 (+8.2%)

(2) The use of multiple query representations significantly improves retrieval performance when compared to equivalent networks with a single query representation.

The results described here also support a number of additional conclusions:

—Conventional probabilistic retrieval strategies based on *tf* and *idf* work well in the network model.

—The use of a nonzero default probability for term belief significantly improves performance.

—The use of query term weights based on the frequency of the term in the query improves performance for natural language queries.

—Even simple Boolean queries perform as well as corresponding probabilistic versions, given the combining functions of Eqs. 1 through 3.

Perhaps the most important conclusion of the current research is that evidence from multiple sources can be combined to improve our estimate of the probability that a document satisfies a query. As a result, the network model provides a natural framework within which to integrate results from several areas of information-retrieval research (e.g., intelligent interfaces or NLP techniques).

REFERENCES

1. CROFT, W. B., AND DAS, R. Experiments with query acquisition and use in document retrieval systems. In *Proceedings of the 13th International Conference on Research and Development in Information Retrieval* Jean-Luc Vidick, Ed. ACM, Sept. 1990, pp. 349-368.
2. CROFT, W. B., AND HARPER, D. J. Using probabilistic models of document retrieval without relevance information. *J. Doc.* 35 (1979), 285-295.
3. CHEESEMAN, P. An inquiry into computer understanding. *Comp. Intell. 4* (Feb. 1988), 58-66.
4. COHEN, P. R., AND KJELDSEN, R. Information retrieval by constrained spreading activation in semantic networks. *Inf. Prccess. Manage. 23*, 2 (1987), 255-268.
5. COHEN, P. R. *Heuristic Reasoning About Uncertainty: An Artificial Intelligence Approach.* Pitman, Boston, Mass., 1985.
6. COOPER, W. S. A definition of relevance for information retrieval. *Inf. Storage Retrieval, 7* (1971), 19-37.
7. CROFT, W. B. A model of cluster searching based on classification. *Inf. Syst. 5*, 3 (1980), 189-195.
8. CROFT, W. B. Boolean queries and term dependencies in probabilistic retrieval models *J. Am. Soc. Inf. Sci. 37*, 2 (1986), 71-77.
9. CROFT, W. B. Approaches to intelligent information retrieval. *Inf. Process. Manage. 23*, 4 (1987), 249-254.
10. CROFT, W. B., AND THOMPSON, R. H. The use of adaptive mechanisms for selection of search strategies in document retrieval systems. In *Proceedings of the ACM/BCS International Conference on Research and Development in Information Retrieval*, C. J. van Rijsbergen, Ed. 1984, pp. 95-110.
11. CROFT, W. B., AND THOMPSON, R. H. *I^3R*: A new approach to the design of document retrieval systems. *J. Am. Soc. Inf. Sci., 38* (Nov. 1987), 389-404.
12. CROFT, W. B., AND TURTLE, H. A retrieval model incorporating hypertext links. In *Hypertext '89 Proceedings* 1989, pp. 213-224.
13. DEMPSTER, A. P. A generalization of Bayesian inference. *J. Royal Stat. Soc. B.* (1968), 205-247.
14. DOYLE, J. A truth maintenance system. *Artif. Intell. 12*, 3 (1979), 231-272.
15. FURNAS, G. W., LANDAUER, T. K., GOMEZ, L. M., AND DUMAIS, S. T. The vocabulary problem in human-system communication. *Commun. ACM, 30*, 11 (Nov. 1987), 964-971.
16. FOX, E. A., NUNN, G. L., AND LEE, W. C. 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* (Grenoble, June 13-15, 1988). ACM, New York, 1988, pp. 291-308.
17. FOX, E. A. Characterization of two new experimental collections in computer and information science containing textual and bibliographic concepts. Technical Report 83-561. Dept. of Computer Science, Cornell Univ., Ithaca, N.Y., Sept. 1983.
18. FURH, N. Models for retrieval with probabilistic indexing. *Inf. Process. Manage. 25* 1 (1989), 55-72.
19. KANAL, L. N., AND LEMMER, J. F., EDS. *Uncertainty in Artificial Intelligence.* North-Holland, Amsterdam, 1986.
20. KATZER, J., MCGILL, M. J., TESSIER, J. A., FRAKES, W., AND DASGUPTA, P. A study of the overlap among document representations. *Inf. Technol. Res. Dev. 1* (1982), 261-274

21. LEMMER, J. F., AND KANAL, L. N., EDS. *Uncertainty in Artificial Intelligence 2*. North-Holland, Amsterdam, 1988.
22. LAURITZEN, S. L., AND SPIEGELHALTER, D. J. Local computations with probabilities on graphical structures and their application to expert systems. *J. Royal Stat. Soc. B*, 50 2 (1988), 157-224.
23. MARON, M. E., AND KUHNS, J. L. On relevance, probabilistic indexing and information retrieval. *J. ACM*, *7* (1960), 216-224.
24. MCGILL, M., KOLL, M., AND NOREAULT, T. An evaluation of factors affecting document ranking by information retrieval systems. Tech. Rep., Syracuse Univ , School of Information Studies, 1979.
25. NILSSON, N. J. Probabilistic logic. *Artif Intell. 28*, 1, (1986), 71-87
26. NOREAULT, T., KOLL, M., AND MCGILL, M. J. Automatic ranked output for Boolean searches in SIRE *J. Am Soc Inf Sci. 28*, 6 (1977), 333-339.
27. ODDY, R. N., PALMQUIST, R. A., AND CRAWFORD, M. A. Representation of anomalous states of knowledge in information retrieval. In *Proceedings of the 1986 ASIS Annual Conference*. 1986, pp. 248-254.
28. PEARL, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* Morgan Kaufmann Publishers, 1988.
29. ROBERTSON, S. E. The probability ranking principle in IR *J. Doc. 33*, 4 (Dec. 1977), 294-304.
30. SALTON, G. A simple blueprint for automatic Boolean query processing. *Inf. Process Manage. 24*, 3 (1988), 269-280
31. JONES, K. S., AND BATES, R. G Research on automatic indexing 1974-1976. Tech. Rep. Computer Laboratory, Univ. of Cambridge, 1977.
32. SALTON, G., AND BUCKLEY, C. Term weighting approaches in automatic text retrieval *Inf. Process. Manage. 24*, 5 (1988), 513-523
33. SALTON, G., FOX, E., AND WU, H. Extended Boolean information retrieval. *Commun. ACM*, *26*, 11 (Nov. 1983), 1022-1036.
34. SHAFER, G. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, N J., 1976.
35. SIEGEL, S. *Non-parametric Statistics for the Behavorial Sciences*. McGraw-Hill, New York, 1956.
36. SALTON, G , AND MCGILL, M J. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.
37. STIRLING, K. H. The effect of document ranking on retrieval system performance: A search for an optimal ranking rule. *Proc. Am. Soc Inf. Sci. 12* (1975), 105-106.
38. THOMPSON, R. H., AND CROFT, W. B. Support for browsing in an intelligent text retrieval system. *Int. J. Man-Mach. Stud.*, *30* (1989), 639-668.
39. TURTLE, H., AND CROFT, W. B. Efficient evaluation for probabilistic retrieval. In *RIAO91 Conference Proceedings* (Barcelona, Apr. 3-5, 1991), pp. 644-661.
40. TONG, R. M., AND SHAPIRO, D. Experimental investigations of uncertainty in a rule-based system for information retrieval *Int. J. Man-Mach. Stud. 22* (1985), 265-282.
41. TURTLE, H. *Inference Networks for Document Retrieval*. PhD thesis, Computer and Information Science Dept., Univ. of Massachusetts, Amherst, Mass., 1990. Available as COINS Tech. Rep. 90-92.
42. VAN RIJSBERGEN, C. J. *Information Retrieval* Butterworths, London, 1979.
43. VAN RIJSBERGEN, C. J. A non-classical logic for information retrieval. *Comput. J. 29*, 6 (1986), 481-485.
44. WILSON, P. Situational relevance. *Inf. Storage Retrieval 9* (1973), 457-471.
45. ZADEH, L. A. The role of fuzzy logic in the management of uncertainty in expert systems *Fuzzy Sets Sys. 11*, (1983), 199-228.

Received January 1991; revised January 1991; accepted February 1991