Keyword Weight Propagation for Indexing Structured Web Content *

Jong Wook Kim
Comp. Sci. and Eng. Dept.
Arizona State University, Tempe, AZ 85287
jong@asu.edu

K. Selcuk Candan
Comp. Sci. and Eng. Dept.
Arizona State University, Tempe, AZ 85287
candan@asu.edu

ABSTRACT

When documents are *atomically structured*, it is possible to assign them keyword vectors to support indexing. Most web content, however, have non-atomic structures. These include navigational/semantic hierarchies on the web. Although they are especially effective for browsing, such structures make it hard for individual nodes to be properly indexed. This is because, in many cases, their contents have to be *inferred* from the contents of their neighbors, ancestors, and descendants in the structure. In this paper, we propose a novel keyword and keyword weight propagation technique to properly enrich the data nodes in structured content. In particular, our approach first relies on understanding the context provided by the relative content relationships between entries in the structure. We then leverage this information for relative-content preserving keyword propagation. Experiments show that we observe a significant improvement ($10-15\%$) in precision with the proposed keyword propagation algorithm.

1. INTRODUCTION

Many web sites and portals organize their content in terms of navigation hierarchies. These include structurally simple hierarchies, such as *Yahoo.com* [1] , as well as more complex web sites and portals. Naturally, in most cases, navigation hierarchies reflect the semantic relationships (such as generality and specialization) between the data content. In complex portals, for instance, nodes closer to the root are more general than those deeper in the navigation hierarchy (e.g., Figure 1). Although these are effective when users are browsing with the aim of locating a specific content, they provide challenges for effective indexing. Since many of the intermediate nodes are there simply to cluster relevant material together and redirect the users to the lower levels in the hierarchy, the keyword contents of these nodes themselves do not describe *unambiguously* their places in the underlying semantic hierarchy (Figure 1). As a result keyword searches are usually directed to the root of the hierarchy (causing an undesirable topic drift) or to the leaves (which individually may not be enough to satisfy the query). Thus, it is important to identify the appropriate intermediary node/page in the hierarchy to direct the user.

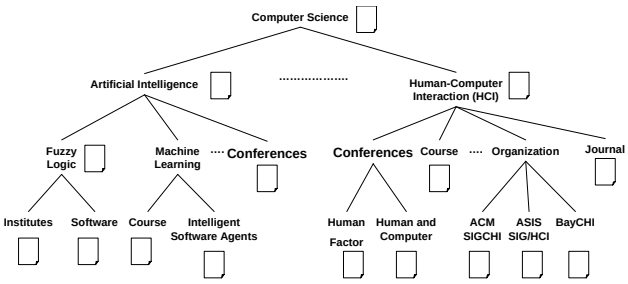


Figure 1: The Yahoo CS hierarchy: the contents of the individual nodes do not state unambiguously their places in the semantic hierarchy (e.g.,"Conferences").

To address this shortcoming, in this paper *we develop a keyword and keyword weight propagation technique*[1]*, which exploits the context provided by the relative contents of the neighboring entries to propagate keywords along a structure to enrich the individual nodes for effective searches.* This keyword propagation technique enables keyword based indexing of hierarchical web data.

Given a navigation hierarchy, a naive solution to keyword propagation could be to treat an entire sub-structure (the node and all the descendant nodes) as a single atomic document and use all the corresponding keywords. However, this naive solution does not necessarily associate the correct set of keywords with the intermediary nodes as it ignores the fact that a node can inherit content both from its ancestors in the hierarchy and its descendants.

As one moves up in the navigational hierarchy, the nodes get more *generalized* and subsume the content of their descen-

*This work is supported by an NSF ITR Grant, ITR-0326544; "*ILearn: IT-enabled Ubiquitous Access for Educational Opportunities for Blind Individuals*" and an RSA Grant "*Ubiquitous Environment to Facilitate Access to Textbooks and Related Materials for Adults and School Age Children who are Blind or Visually Impaired*".

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
WEBKDD'06 August 20, 2006, Philadelphia, Pennsylvania, USA.
Copyright 2006 ACM 1-59593-444-8...$5.00

[1]In the rest of the paper, we use the terms "keyword weight propagation" and "keyword propagation" interchangeably.

dant nodes. As one moves down in a navigational/semantic hierarchy, the nodes get more *specialized* and focus on specific content. It is important that the keyword content of the nodes reflect this specialization/generalization behavior, while still acting as appropriate discriminators. For instance, in Figure 1, the nodes labeled "Artificial Intelligence" and "Human Computer Interaction" are good discriminators, but their children labeled "Conferences" may not be if the parent labels are not appropriately reflected in the children nodes. A pertinent question is whether the children should inherit keywords from their parents or whether parents should inherit keywords from their children. We propose that a node should inherit content from both its ancestor as well as its descendants in the hierarchy; except that the keyword inheritance should not cause the flattening of the semantic hierarchy. In other words, while keywords and keywords weights are inherited, the original semantic structure (i.e., generalization/specialization) should be preserved. The challenge of course is to be quantify the degrees of generalization and specialization inherent in the original structure and to determine the appropriate degree of keyword inheritance in a way that preserves the original semantic structure inherent in the hierarchy.

1.1 Contributions of the Paper

Based on the above observations, we propose a keyword-propagation algorithm which leverages the relationships between the entries in a structure. In particular,

- we develop a method for discovering and *quantifying* the inherent generalization/specialization relationship between entries in a navigation hierarchy, based on their relative content (Section 3.1),
- we introduce a novel keyword propagation algorithm, which uses this relationship between a pair of neighboring entries to propagate keywords and keyword weights between them (Section 4.1), and
- we show how to extend this pairwise technique to the indexing of more complex structures (Section 4.2).

We then experimentally evaluate the proposed techniques in Section 5. Results show that the proposed techniques provide a $10-15\%$ improvement in precision and capture the domain context better than alternative solutions.

2. RELATED WORK

Various techniques have been proposed to leverage the web structure in identifying document associations, such as the companion and co-citation algorithms proposed by Dean and Henzinger [11]. One approach to organizing web query results based on available web structure is *topic distillation* proposed in [19]. The basic idea in topic-distillation and related work is to use random walks on the structure of the Web to propagate scores between pages in a way to organize topic spaces in terms of smaller sets of hub and authoritative pages. PageRank [6] computes single scores as opposed to hub and authority scores. [22] applied random walk approach to solve the reverse problem of *"finding on which topic a given page is reputable"*. A related problem of *"finding why given two pages are related"* is addressed by [7, 8] again using a random walk based score propagation. Unlike these works, instead of basing score propagation on a "random surfer", we aim to understand the semantic structure of the web content and preserve it during keyword weight propagation.

Bharat and Henzinger [5] showed that the hubs and authorities having the high score may deviate from the original topic, which is called topic drift. To prevent the topic drift in multitopic pages, Chakrabarti et al. [9] presented an algorithm which accumulates the hub score along the frontier nodes. These techniques are usually general purpose and are not explicitly designed to leverage the semantically-supported structures of navigation hierarchies.

2.1 Score and Keyword Frequency Propogation

Recently, there has been growing research on integrating keyword search into graph-structured databases. BANKS [4] tries to find the rooted subtree connecting all the nodes that match a keyword in the query. It proposes a *backward expansion* search that starts from the nodes matching keyword in the query and finds a common node from which there exist paths to the starting nodes. Kacholia et al. [15] introduces a *bidirectional expansion* search method, where besides backward search, forward search is done from nodes that might be potential root nodes. XRank [14] and ObjectRank [3] propose algorithms, similar to PageRank [6], to compute ranking results for keyword-style database queries. XSEarch [10], a semantic search engine for XML data, relies on extended information retrieval techniques in ranking results.

Unlike the above approaches which propagate scores representing structural properties of the pages, other methods propagate the term frequency values or (given a query) the relevance score itself. For instance, given a query [29] propagates the relevance score between web pages connected with hyperlinks. On the other hand, the term frequency values can be propagated between neighboring pages [30, 27]. Recently, Qin et al. [20] proposes a generic relevance propagation framework, which brings together techniques from [29] and [30]. The approach we present in this paper is also a term-frequency-propagation technique; however, unlike the above methods, the techniques we introduce leverage the context provided by entries in the structures.

Hierarchical classification is the process of categorizing unlabeled data into a hierarchical structure. The purpose of hierarchical classification of web data is to minimize error rate by assigning them into intermediate level in the hierarchy instead of categorizing them into leaf level. It leverages content from ancestors and descendants to reduce errors when assigning data [31, 12]. Even if these approaches are promising in web categorization, they are not directly applicable to indexing as they need training data.

2.2 Measuring the Degree of Generalization/ Specialization in an Hierarchy

There have been a number of proposals for measuring semantic similarities between keywords in a taxonomy. These approaches can be classified into two categories: structure-based or information-based methods. [21] proposes that the conceptual distance between two concept-nodes should be defined as the shortest path between two nodes in the tax-

onomy and that this should satisfy metric distance properties. This approach proved to be very useful in small and specific domains, such as medical semantic nets. However, it ignores that (a) the semantic distance between neighboring nodes are not always equal and that (b) the varying local densities in the taxonomy can have strong impacts on the semantic distance between concept-nodes. To overcome these shortcomings, [24] associate weights to the edges in the hierarchy: the edge weight is affected both by its depth in the hierarchy and the local density in the taxonomy. To capture the effect of the depth, [32] estimates the conceptual distance between two concepts, c_1 and c_2, by counting the number of edges between them, and normalizing this value using the number of edges from the root of the hierarchy to the closest common ancestor of c_1 and c_2.

Note that the above approaches try to measure the semantic relationships in a given concept hierarchy, without the help of any supporting text associated to each node. We, on the other hand, note that pages in navigation hierarchies and portals contain supporting text that can be leveraged for more informed measurement of generalization and specialization. Thus, we next introduce the concept of *relative content* of entries in a navigation structure.

3. RELATIVE CONTENT OF ENTRIES

As discussed in the introduction, our goal in this paper is to develop a keyword propagation technique, which leverages the context inherent in navigation structures. Naturally, to implement such a technique, we first have to identify and quantify the semantic relationship between a given node and its neighbors. In a navigation hierarchy, the semantic relationships between various entries are usually in the form of *specializations* and *generalities* [17]:

- A specialized entry corresponds to a more focussed (or constrained) concept, while
- a general entry is less constrained.

For instance, in the CS hierarchy shown in Figure 1, "Machine Learning" is more *general* than its child "Intelligent Software Agents", but more *specialized* than its parent "Artificial Intelligence". Thus, discovering the degree of semantic relationship between two neighboring nodes in a structure involves identifying the *specialization* and *generality* direction and degrees between these two nodes.

In semantically meaningful structures, such as `IS-A` hierarchies in ontologies, the direction of generalization/specialization usually implied by the parent/child (ancestor/descendant relationships): an entry closer to the root is more general than a descendant deeper in the hierarchy. However, quantifying *how general* one entry is relative to another one is not trivial. Identifying semantic similarity/dissimilarity between entries in a semantic hierarchy is a well studied problem [23, 21, 24, 18, 16]. Most existing techniques extract these relationships either (a) from the information content of the terms in an ontology (computed over a large corpus) [23, 18, 16] or (b) from the structure (eg., density, depth) in the hierarchy itself [21, 24]. In this paper, we note that, when we consider hierarchies of text or navigation structures, there is a third and complementary piece of information (i.e., the textual content of the entries themselves) which can be exploited to extract the relationships between entries.

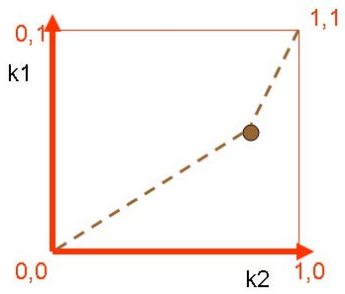


Figure 2: An entry with two keywords, k_1 and k_2

In this section, we discuss how to measure the relative content of a pair of parent/child entries.

3.1 Intuition

Let us consider two entries, A and B, in a navigation hierarchy. Let us further assume that keyword vectors, $\vec{A}$ and $\vec{B}$, represent the node contents quantitatively. Depending on which data representation model is used, the keyword weights in these vectors could be TF/IDF values or some other feature importance measures.

One way to think of relative content of these two data entries is in terms of constraints imposed on them by their keyword compositions: the statement that "*entry A is more general than B*" can be interpreted as A being less constrained than B by its keywords.

EXAMPLE 3.1. *Let us consider two entries, A and B, where A is an ancestor of B in the navigation hierarchy. Let us assume that A has three independent keywords, k_1, k_2 and k_3, and B has only two keywords, k_2 and k_3:*

- *Since the ancestor, A, is more general than the descendant, B, in the navigation hierarchy, the extra keyword, k_1, must render A less constrained. In a sense,* **if** *B is interpreted as $k_2 \vee k_3$,* **then** *A should be interpreted as $k_1 \vee k_2 \vee k_3$ (less constraining than $k_2 \vee k_3$).*

In order to be able to benefit from this observation in measuring degrees of generality, we need to be able to *quantify* how well a given entry can be interpreted as the disjunction of the corresponding keywords.

3.2 Quantifying the Degree of "OR"ness of a Keyword Vector

Extended boolean model [25] of vector spaces associates well defined disjunctive and conjunctive semantics to document in order to be able to answer boolean queries on vector data. For example, consider an entry, e, with two keywords k_1 and k_2 in Figure 2. The extended boolean model interprets the point $(1, 1)$ as a (hypothetical) entry which contains both k_1 and k_2; i.e, $k_1 \wedge k_2$; in contrast, the origin, $\vec{\mathcal{O}} = \langle 0, 0 \rangle$ is interpreted as an entry which contains neither k_1 nor k_2; i.e, $\neg k_1 \wedge \neg k_2 = \neg(k_1 \vee k_2)$. Thus, in extended boolean

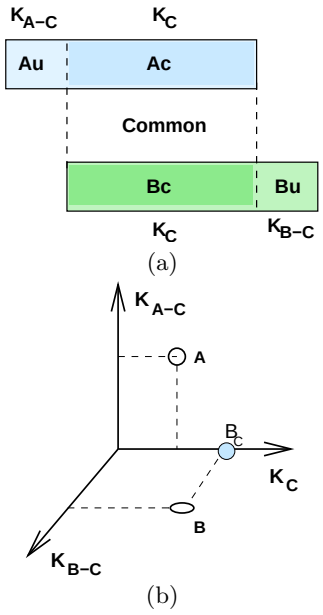


Figure 3: (a) Visual representation of the keyword contents of two data entries: they share a common keyword base, but they also have their own individual keyword content; (b) vector space representation of A, B, and B_C (B projected onto the common keyword space)

model, the and-ness of this entry can be measured as its similarity to the entry $(1, 1)$; i.e., an entry closer to $(1, 1)$ better matches the statement $k_1 \wedge k_2$ than an entry further away. The or-ness of an entry, on the other hand, can be measured as its dissimilarity from the entry $(0, 0)$; in other words, an entry further away from $(0, 0)$ better matches the statement $k_1 \vee k_2$ than an entry further away.

We can apply this observation to our example, Example 3.1, as follows:

EXAMPLE 3.2. *In Example 3.1, let the origin point, $\vec{\mathcal{O}} = \langle 0, 0, 0 \rangle$, correspond to a (hypothetical) entry which does not contain k_1, k_2 and k_3. In other words, $\mathcal{O}$ can be interpreted as $(\neg k_1 \wedge \neg k_2 \wedge \neg k_3)$ or equivalently as $\neg(k_1 \vee k_2 \vee k_3)$. Since $\mathcal{O}$ corresponds to $\neg(k_1 \vee k_2 \vee k_3)$, how much a entry A represents a disjunct can be measured by $|\vec{A} - \vec{\mathcal{O}}|$; i.e., the length, $|\vec{A}|$, of the vector $\vec{A}$. In a similar way, $|\vec{B}|$ (in the same space) measures the degree of or-ness for entry B, consisting of keywords k_2 and k_3.*

If A is said to be more general than B, then $|\vec{A} - \vec{\mathcal{O}}|$ needs to be larger than $|\vec{B} - \vec{\mathcal{O}}|$.

Note that, although this example illustrates the special case where one entry contains three keywords (k_1, k_2, k_3) and the other contains two (k_2, k_3) of these three, it also illustrates the underlying mechanism we will exploit to evaluate the relative degree of generality between a pair of entries.

3.3 Evaluating the Relative Generality

As shown in Figure 3(a), given A and B in the navigation hierarhcy, we can split each one into two fragments (Figure 3): the two entries will have a common keyword base, K_C[2] (set of keywords). The entries will also have their own individual sets of keywords, K_{A-C} and K_{B-C}. We denote the fragments of the entries, A and B, corresponding to the common base K_C, as A_C and B_C (and corresponding keyword vectors, $\vec{A_C}$ and $\vec{B_C}$), respectively. We also denote the un-common fragments of the two entries, as A_U and B_U (and corresponding keyword vectors, $\vec{A_U}$ and $\vec{B_U}$[3]). Thus, these two entries (and their fragments) can be represented as vectors in a vector space, where the dimensions correspond to keyword sets K_{A-C}, K_{B-C}, and K_C, as shown in Figure 3(b)[4].

DEFINITION 3.1 (RELATIVE CONTENT). *Given two entries, A and B, we can define the degree of* generality *of A relative to (the common base of A and) B as*

$$R_{AB} = \frac{|\vec{A}|}{|\vec{B_C}|} = \frac{|\vec{A_U} + \vec{A_C}|}{|\vec{B_C}|}.$$

This essentially measures whether the additional keywords $\vec{A_U}$ that do not appear in B render A more general or less general (more specialized) than $\vec{B_C}$. We also refer this as the relative content *of A vs. (the common base of A and) B. R_{BA} is similarly defined.*

Note that, when available, additional information-theoretic [23, 18, 16] and structural information [21, 24] (commonly used for evaluating semantic similarity when nodes do not have text associated with them) can also be used in supporting the degree of generality and the relative content values computed based on the text associated to the nodes in the navigation hierarchy. In this paper, we do not consider such extensions.

4. KEYWORD PROPAGATION

Proper keyword-based retrieval of entries that fall into a semantically-based structure requires identification of the appropriate keywords representing each entry in this structure. When the entries themselves do not originally contain all appropriate keywords, then these keywords (and keyword weights) need to be propagated among the entries.

In this section, we first discuss how to propagate keywords (and their weights) between a neighboring pair of entries using the knowledge of the relative content relationship between them. Note that, for our purposes, it is sufficient to concentrate initially only on parent/child entries as the keyword propagation in the hierarchy will follow edges between individual parent/child pairs. Thus, we can extend the keyword propagation between parent/child pairs to the propagation of keywords in more complex structures as described in Section 4.2.

[2] C stands for *common*.
[3] U stands for uncommon.
[4] Note that, although this figure represents the vectors in a 3-dimensional space for ease of visualization, common and un-common bases are in reality collections of keywords.

4.1 Relative Content Preserving Keyword Propagation between a Pair of Entries

The main purpose of keyword propagation is to enrich the entries in a navigation hierarchy by keywords inherited from their neighbors. Naturally, such a propagation of keywords should change the composition of the keyword vectors of the entries, yet *after the keyword propagation the semantic hierarchy should not end up being be flattened due to the shared keywords*; thus, the original semantic properties (i.e., relative generality) of the pair of entries should be preserved, despite the fact that there is larger keyword sharing between the entries.

Based on this observation, we compute a pairwise propagation degree between two neighboring entries as follows:

DEFINITION 4.1 (PAIRWISE PROPAGATION DEGREE, α). *Given two entries, A and B, where a_i is the weight associated with keywords $k_i \in K_A$ and b_j is the weight for $k_j \in K_B$, the degree of pairwise keyword propagation is a parameter, α, such that, after the propagation, we obtain two enriched entries, A' and B', such that*

- *for all $k_i \in K_{A'}$,*
  - *if $k_i \in (K_A - K_B)$, then $a'_i = a_i$,*
  - *if $k_i \in (K_A \cap K_B)$, then $a'_i = a_i + \alpha b_i$, and*
  - *if $k_i \in (K_B - K_A)$, then $a'_i = \alpha b_i$*
- *similarly, for all $k_i \in K_{B'}$,*
  - *if $k_i \in (K_B - K_A)$, then $b'_i = b_i$,*
  - *if $k_i \in (K_B \cap K_A)$, then $b'_i = b_i + \alpha a_i$, and*
  - *if $k_i \in (K_A - K_B)$, then $b'_i = \alpha a_i$.*

Note that after the keyword propagation, these two entries are located in a common keyword space, K_C, where

$$K_C = K_{A'} = K_{B'} = K_A \cup K_B.$$

In other words, the keyword propagation degree, α, governs how much content these two pairs of entries will exchange. Naturally, if we would like for the keyword propagation to be relative content preserving, we need to ensure that

$$R_{A'B'} = R_{AB}.$$

Since after enrichment K_C will be equal to $K_{B'}$, we can restate this requirement as

$$R_{AB} = \frac{|\vec{A}|}{|\vec{B_C}|} = \frac{|\vec{A'}|}{|\vec{B'}|} = R_{A'B'}$$

Thus, we compute the appropriate propagation degree, by solving the resulting equation for α.

4.1.1 When to Apply Propagation?

One critical observation is that keyword propagation is applicable only when A and B have at least one uncommon keyword (i.e., $K_A \neq K_B$). To see this, let us also consider the special case where the two document entries A and B have originally the same keyword content (i.e., $K_A = K_B = K_C$ and $K_{A-C} = K_{B-C} = \emptyset$). In this particular case,

$$R_{AB} = \frac{|\vec{A}|}{|\vec{B_C}|} = \frac{|\vec{A}|}{|\vec{B}|} = \left(\frac{\sum_{k_i \in K_A = K_C} {a_i}^p}{\sum_{k_i \in K_A = K_C} {b_i}^p}\right)^{\frac{1}{p}},$$

and

$$R_{A'B'} = \frac{|\vec{A'}|}{|\vec{B'}|} = \left(\frac{\sum_{k_i \in K_{A'} = K_C} (a_i + \alpha b_i)^p}{\sum_{k_i \in K_{A'} = K_C} (b_i + \alpha a_i)^p}\right)^{\frac{1}{p}}.$$

If we try to solve for $R_{AB} = R_{A'B'}$, we will get $\alpha = 0$ as a solution. Consequently, keyword propagation is not applicable when two entries, A and B, have the same set of keywords. As discussed in the next section, this provides a stopping condition for the keyword propagation process on more complex structures.

4.2 Keyword Propagation across a Complex Structure

By propagating keywords between a given pair of entries, we enable both entries to get enriched based on their contexts relative to each other. However, in a more complex structure, considering only pairs of contents is not sufficient. Therefore, the propagation process should not be limited to pairs of entries.

4.2.1 Propagation Adjacency Matrix

Let $\mathcal{H}(N, E)$ denote a navigation hierarchy, where N is the set of nodes (corresponding to the entries in the hierarchy) and E is the set of edges (corresponding to the navigational links) between these pairs of nodes in N.

DEFINITION 4.2 (PROPAGATION ADJACENCY MATRIX). *Given a navigation hierarchy, $\mathcal{H}(N, E)$, its corresponding propagation adjacency metric, $\mathtt{M}$, is defined as follows:*

- *if there is an edge $e_{ij} \in E$, then both (i, j) and (j, i) of $\mathtt{M}$ is equal to α_{ij} (i.e., the pairwise propagation degree between the corresponding nodes);*
- *otherwise, both (i, j) and (j, i) of $\mathtt{M}$ are equal to 0.* $\diamond$

Note that $\mathtt{M}$ is symmetric and the diagonal values of $\mathtt{M}$ are equal to 0.

4.2.2 Keyword Propagation Operator

Given a navigation hierarchy, $\mathcal{H}(N, E)$, let $\mathtt{T}$ be the corresponding term-node (or term-document) matrix, where (t, i) of $\mathtt{T}$ is the weight of term k_t in node $n_i \in N$.

Furthermore, let $\mathtt{P}$ denote the term propagation matrix, where (t, i) is the propagation weight of term k_t in node n_i being inherited from its neighbors in the adjacency matrix, $\mathtt{M}$. Thus, the keyword propagation matrix, $\mathtt{P}$, can be computed as follows:

DEFINITION 4.3 (KEYWORD PROPAGATION OPERATOR). *The keyword propagation operator, $\oplus$, is such that, given a*

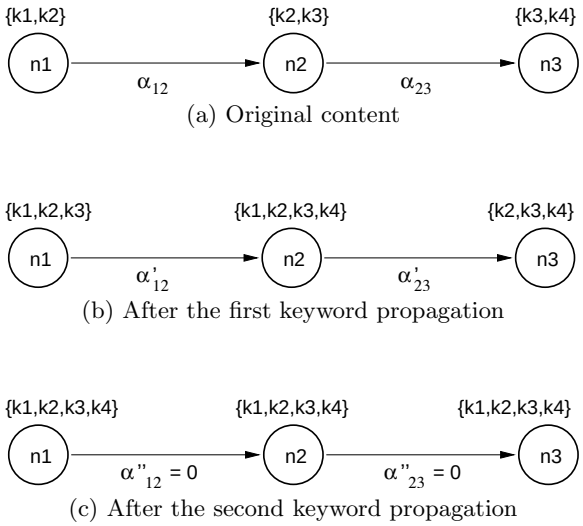


Figure 4: Keyword propagation across a structure: since the maximum distance between any pair of nodes is two, it takes two propagation cycles to propagate all the keywords across the structure; note that after the last phase, the value of the pairwise keyword propagation degree for all edges is 0, and the process stops

term-node matrix $\mathtt{T}$ *and a propagation adjacency matrix* $\mathtt{M}$, $\mathtt{P} = \mathtt{T} \oplus \mathtt{M}$ *is the keyword propagation matrix. Note that, based on the definition of pairwise propagation, the entry (t,i) of* $\mathtt{P}$ *is computed as follows:*

$$\mathtt{P}_{t,i} = \sum_h \mathtt{T}_{t,h} \times \mathtt{M}_{i,h}.$$

Furthermore, since $\mathtt{M}$ *is symmetric,*

$$\mathtt{P}_{t,i} = \sum_h \mathtt{T}_{t,h} \times \mathtt{M}_{h,i}.$$

Thus, we can see that the semantics of the keyword propagation operator is that of matrix multiplication:

$$\mathtt{P} = \mathtt{T} \oplus \mathtt{M} = \mathtt{TM},$$

where $\mathtt{TM}$ *is the matrix multiplication of* $\mathtt{T}$ *and* $\mathtt{M}$.

Thus, the new *enriched* term-node matrix is equal to

$$\mathtt{T}' = \mathtt{T} + \mathtt{P} = \mathtt{T} + (\mathtt{T} \oplus \mathtt{M}) = \mathtt{T} + \mathtt{TM} = \mathtt{T}(\mathtt{I} + \mathtt{M}) = \mathtt{TM_I}$$

where $\mathtt{I}$ is the identity matrix. Note that since all diagonal values in $\mathtt{M}$ are zero, $\mathtt{M_I} = \mathtt{I} + \mathtt{M}$ is such that all diagonal values are 1 and all non-diagonal entries are those in $\mathtt{M}$.

4.2.3 Keyword Propagation Process

After the application of $\oplus$, each node in the structure has not only its original keywords, but also keywords inherited from its immediate neighbors. Therefore, after the first application of the keyword propagation operator, $\mathtt{T}' = \mathtt{TM_I}$ is the first-level enrichment ($\mathtt{T_1}$) of $\mathtt{T}$.

Since, during propagation, at least one node inherited keywords from its neighbors, the columns of $\mathtt{T_1}$ have fewer zeros than the columns of $\mathtt{T}$. This process is visualized in Figures 4 (a,b,c) with an example:

1. Initially, the three nodes in the example each have a different keyword set (Figure 4 (a));
2. After the first keyword propagation step, the middle node (which has two neighbors) has all possible keywords, whereas the left- and the right-most nodes did not receive each others' keywords yet (Figure 4(b)).
3. Thus, the process has to be repeated (Figure 4 (c)).
4. After the second iteration, all nodes received all keywords (but with different weights). Since the keyword sets of all the nodes are the same, the pairwise propagation degrees are all 0s, and the keyword propagation process stops.

Note that, since at each iteration the keyword composition of the nodes changes, the pairwise keyword propagation degrees may need to be recomputed in a way to preserve the relative generality values.

We can generalize this process as follows: Let d be the diameter (the greatest number of edges between any nodes) in the navigation hierarchy, $\mathcal{H}$. Then, the final enriched term-node matrix is equal to

$$\mathtt{T}_{final} = \mathtt{TM_{I1}M_{I2}M_{I2}} \ldots \mathtt{M_{Id}},$$

where $\mathtt{M_{Im}}$ is the propagation adjacency matrix computed for the m^{th} iteration. Since, after the d^{th} iteration, the keyword sets of all nodes in the structure are going to be the same, all pairwise keyword propagation degrees are 0. Thus, the process naturally stops after d steps.

5. EXPERIMENTS

We evaluated the effectiveness of the keyword (weight) propagation technique through a user study. First we describe the experimental setup and then discuss the results.

5.1 Setup

Data: We used Yahoo Computer Science, Mathematics, and Movie directories [1]. *Yahoo Directory* has a structure which follows a class-hierarchy. Each page in the Yahoo directory contains not only the description of itself, but also the description of the web sites listed in the page. The characteristics of the data collected from these directories are shown in Table 1.

Ground truth: We used 10 sample keyword queries, with logical operators *AND* and *OR*, which can be answered using the collected data (Table 2). Given these queries, we established the ground truth by asking 8 users, not involved in the project, to assess the available data relative to the queries. During the ground truth establishment process, assessors were asked to assign them labels: irrelevant, partially-relevant, or relevant. These labels are used to evaluate the relevance (**r**) under three different interpretations: *relaxed*, *differentiated*, and *strict* (Table 3).

Query processing alternatives: The similarity, $sim(n, q)$, between a query q and a data entry, n, is computed using the

extended boolean model [25]. Given a node n, we refer to the node after propagation as n_{kp}. Thus given a query q, the matching score after keyword propagation is $sim(n_{kp}, q)$.

In order to verify that keyword propagation algorithm can indeed successfully capture the implicit context in the hierarchical data, we also used alternative context discovery mechanisms, where the hierarchy is split into domains and keywords for the nodes in each domain are merged (keyword weights are averaged) to create a collective *domain* context. In the experiments, we defined the *domain* of a given node n by using two alternative score propagation schemes:

- *whole subtree as the domain:* in this case, the *domain* of a given node n is defined as the sub-tree that contains node n and is rooted at a child of the root. For example, in Figure 1, *domain* of "Fuzzy Logic" is the subtree, whose root is "Artificial Intelligence"; note that entries "Fuzzy Logic" and "Intelligent Software Agents" have the same *domain*.
- *neighborhood as the domain:* node n obtains its domain context from its parent and children.

Given a node n and a query q, we refer to the similarity between a query q and the *domain* of the node n as $sim(domain(n), q)$. In the experiments, given a node n and a query q we computed the corresponding score using the following alternative methods:

- **case** N**;** *no keyword propagation or other context discovery*: $score(n, q) = sim(n, q)$, i.e., each data node is considered individually without any contextual help.
- **cases** D_t **and** D_n**;** *no keyword propagation, but context extracted from the* domain *is used*: in these cases, similarity of the query to $domain(n)$ is included in varying degrees to the basic node-only similarity; i.e, $score(n, q) = \beta \times sim(n, q) + \gamma \times sim(domain(n), q)$, where $\beta + \gamma = 1$.

  Note that D_t corresponds to the case where domain is represented as the whole subtree and D_n corresponds to the case where the domain of a node is its immediate neighborhood.
- **case** KP**;** *keyword propagation*: in this case, similarity of the query to the node is computed based on keyword propagated version of the data node; i.e., $score(n, q) = sim(n_{kp}, q)$.
- **cases** $KP + D_t$ **and** $KP + D_n$**;** *keyword propagation further augmented with* domain-*provided context*: in these cases, similarity of the query to the node is computed by combining the similarity of the keyword propagated version of the data node, n_{kp}, and the similarity of the domain of the data node: $score(n, q) = \beta \times sim(n_{kp}, q) + \gamma \times sim(domain(n), q)$,, where $\beta + \gamma = 1$.

Evaluation measures: Given the ground truth, in order to evaluate various algorithms, we used (a) **P@10** = $\sum_{i=1}^{10} \frac{\mathbf{r}_i}{10}$, (b) *mean reciprocal rank of the first relevant document* (**MRR**) [28] and (c) **paired t-Test**. MRR is the inverse of the rank of the first relevant document in the ranked list of results [28], and the *paired t-Test* evaluates the significance of the difference between means of two small independent data sets [26].

Table 1: The information of data set

	CS.	Math.	Movie
Average Depth	3.05	3.224	3.514
Maximum Depth	5	5	7
Num. of Categories	107	138	335

Table 3: The relevance weight (r) of labels under three interpretations: Relaxed, Differentiated, and Strict

r	Relaxed	Differentiated	Strict
irrelevant	0	0	0
partially relevant	1	0.5	0
fully relevant	1	1	1

Table 4: P@10 for keyword propagation versus no keyword propagation

	N	KP	Improvement
Relaxed	0.670	0.753	12.27%
Differentiated	0.542	0.612	12.60%
Strict	0.415	0.469	13.10%

Table 5: Average MRR for keyword propagation versus no keyword propagation

	N	KP	Improvement
Relaxed	0.869	0.930	6.97%
Differentiated	0.869	0.930	6.97%
Strict	0.644	0.730	13.20%

Table 6: The p-values from the t-Test are all smaller than 0.05, leading to a $\geq 95\%$ confidence for KP

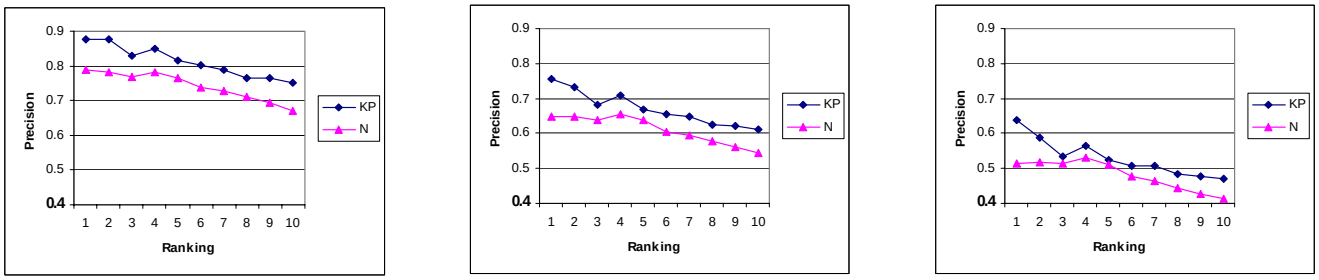
p-values for	Relaxed	Differentiated	Strict
KP vs. N	0.029	0.031	0.047

5.2 Keyword Propagation vs. No Propagation

Table 4 shows the impact of keyword propagation on the precision (β is set to 1 and γ is set to 0). We see that the improvement provided by keyword-propagation over a non-keyword propagation based scheme is around 12-13%, under all three precision measures. Table 5 shows the average MRR for the keyword propagation and no keyword propagation approaches. We see that keyword propagation again improves the MRR values upto 13%. This is especially the case for the strict search, where only those data nodes which were deemed highly similar by the users are counted toward relevance. Table 6 shows the *p-values* computed by comparing KP and N results using the t-Test. The fact that these p-values are all smaller than the baseline value 0.05 indicates that we can be $\geq$ **95**% certain that keyword propagation (KP) functions better than no keyword propagation (N).

Table 2: Sample user queries that we use in the experiments

User Query	Query Representation
Find the information on "*artificial intelligence*"	artificial ∧ intelligence
Find the information on "*evolutionary genetic algorithms*"	evolutionary ∧ genetic ∧ algorithms
Find the information on "*human computer interaction*"	human ∧ computer ∧ interaction
Find the information on "*natural language processing*"	natural ∧ language ∧ processing
Find the information on "*supercomputing center*"	supercomputing ∧ center
Find the information on "*random number or roman number*"	(random ∨ roman) ∧ number
Find the information on "*course or institute of geometry mathematics*"	(course ∨ institute) ∧ geometry ∧ mathematics
Find the information on "*basketball or baseball films in comedy genre*"	(basketball ∨ baseball) ∧ comedy
Find the information on "*parody or horror films in comedy genre films*"	(parody ∨ horror) ∧ comedy
Find the information on "*war films in action or documentary genre*"	(action ∨ documentary) ∧ war



(a) Relaxed Precision vs. Ranking (b) Differentiated Precision vs. Ranking (c) Strict Precision vs. Ranking

Figure 5: Precision provided by the *KP* and *N* algorithms for top ranking results

Figure 5 plots the average precision of the result at various result ranks. The figure shows that keyword propagation consistently improves the precision, at all ranks.

5.3 Keyword Propagation vs. Alternative Context Extraction Mechanisms

We also observed the relative impact of keyword propagation against more traditional domain representation schemes. We varied the β and γ values in the combination as shown in Table 7. The results show that, although alternative context extraction methods are better than nothing, the best results are obtained when keyword propagation is used (and especially, when it is used *alone*).

(D_t/D_n vs. N). Table 7 shows that score propagation (denoted by D_t and D_n) could improve the precision when compared against the case where no context-information is used (denoted by N). With the sub-tree based alternative D_t, the best-result is obtained when the γ value is set to 0.6. Beyond this point, this approach loses the individuality of the nodes and starts under-performing. On the contrary, the best results with the neighborhood-based alternative D_n are obtained when the γ value is set high.

(KP vs. all others). More importantly, Table 7 shows that the **best** results are obtained when keyword propagation is used (KP) instead of score propagation. Furthermore, providing additional domain information ($KP + D$) does not improve the results; in fact, such an augment actually hurts the precision provided by KP. This means that the keyword propagation approach is sufficient (and better than the alternative ways) for capturing data context.

Table 7: KP provides the best results. Additional domain knowledge does not improve KP's result

Relaxed; P@10						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
N	0.670	-	-	-	-	-
D_t	-	0.682	0.683	0.717	0.694	NA
D_n	-	0.658	0.672	0.675	0.699	0.706
KP	**0.753**	-	-	-	-	-
KP+D_t	-	0.752	0.750	0.750	0.733	NA
KP+D_n	-	0.753	0.752	0.736	0.718	0.706

(a) Relaxed P@10

Differentiated; P@10						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
N	0.542	-	-	-	-	-
D_t	-	0.539	0.545	0.579	0.558	NA
D_n	-	0.532	0.542	0.547	0.564	0.572
KP	**0.612**	-	-	-	-	-
KP+D_t	-	0.606	0.607	0.607	0.597	NA
KP+D_n	-	0.611	0.612	0.596	0.584	0.572

(b) Differentiated P@10

Strict; P@10						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
N	0.415	-	-	-	-	-
D_t	-	0.396	0.407	0.439	0.422	NA
D_n	-	0.405	0.412	0.419	0.429	0.436
KP	**0.469**	-	-	-	-	-
KP+D_t	-	0.461	0.464	0.464	0.460	NA
KP+D_n	-	0.469	0.469	0.457	0.449	0.436

(c) Strict P@10

Table 8: Confidence associated with performance (relative to N, no-keyword propagation) evaluated by t-Test

Differentiated; t-Test						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
D_t vs. N	-	worse	55.1%	84.4%	63.5%	NA
D_n vs. N	-	worse	54.0%	65.2%	81.1%	90.5%
KP vs. N	**96.9**%	-	-	-	-	-
KP+$D_t$vsN	-	96.2%	95.7%	95.7%	90.0%	NA
KP+$D_n$vsN	-	96.7%	96.8%	91.8%	86.5%	90.5%

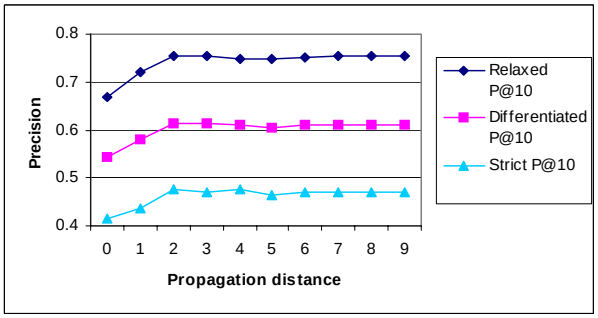


Figure 6: The effect of propagation distance: there is a significant improvement at 1- and 2-distances; the performance becomes stable after 3-distance.

Finally, Table 8 verifies these results using the alternative t-Test metric for differentiated interpretation of relevance labels (results for the other interpretations are similar). As this table shows, *KP alone provides the largest confidence value when compared to N, i.e., no keyword propagation.*

5.4 Effect of the Structural Distance

In order to observe the effects of the distances between the entries in the structure, we artificially varied the maximum allowed propagation distance. Results are reported in Figure 6: the first value in the curves corresponds to 0-distance (no propagation); the next value (1-distance) corresponds to the immediate neighbor propagation, and so on. As can be seen in this figure, we observed significant improvements at both 1-distance and 2-distance propagation cases. Beyond 3-distance the precision becomes stable. This means that keyword propagation improves the precision, but when the entries in the structures are far apart, their impacts on each others are not discernable.

5.5 Statisitical Validation of the Ground Truth

Finally, we ran a statistical test to observe the agreement between the assessors used for creating the ground truth for the experiments. For this purpose, we ran an ANOVA test [13], which measures the difference between the means of two or more groups (evaluator label judgments in our case). The ANOVA test provides an F value which needs to be smaller than an $F_{critical}$ value to ensure that none of the observations is significantly different from the others. In our experiments, the F value returned by the ANOVA test was 6.89 against a $F_{critical}$ value of 2.13 for 95% confidence. This implies there was a disagreement among evaluator label judgments. In particular, we identified two evaluators whose judgments were significantly different from the others.

Table 9: When the judgements provided by the core group is considered, KP still provides the best precision

Relaxed; P@10						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
N	0.652	-	-	-	-	-
D_t	-	0.674	0.680	0.717	0.698	NA
D_n	-	0.636	0.652	0.659	0.665	0.697
KP	**0.762**	-	-	-	-	-
KP+D_t	-	0.759	0.758	0.758	0.737	NA
KP+D_n	-	0.757	0.755	0.744	0.727	0.697

(a) Relaxed P@10

Differentiated; P@10						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
N	0.538	-	-	-	-	-
D_t	-	0.547	0.556	0.594	0.571	NA
D_n	-	0.525	0.537	0.544	0.565	0.573
KP	**0.628**	-	-	-	-	-
KP+D_t	-	0.625	0.624	0.624	0.608	NA
KP+D_n	-	0.625	0.625	0.614	0.601	0.573

(b) Differentiated P@10

Strict; P@10						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
N	0.415	-	-	-	-	-
D_t	-	0.420	0.432	0.455	0.445	NA
D_n	-	0.414	0.422	0.428	0.434	0.450
KP	**0.505**	-	-	-	-	-
KP+D_t	-	0.487	0.488	0.488	0.480	NA
KP+D_n	-	0.495	0.496	0.484	0.495	0.450

(c) Strict P@10

Table 10: Confidence associated with performance (relative to N) evaluated by t-Test with the judgements provided by the core group

Differentiated; t-Test						
β/γ	1/0	0.8/0.2	0.6/0.4	0.4/0.6	0.2/0.8	0/1
D_t vs. N	-	62.0%	71.3%	80.4%	72.1%	NA
D_n vs. N	-	worse	worse	69.9%	74.9%	91.8%
KP vs. N	**97.3**%	-	-	-	-	-
KP+$D_t$vsN	-	96.4%	96.6%	96.6%	90.5%	NA
KP+$D_n$vsN	-	96.4%	96.4%	93.8%	90.5%	91.8%

When the ANOVA test was rerun excluding these two evaluators, the new F value was computed as 1.55 against an $F_{critical}$ value of 2.39, highlighting that the evaluator judgments for the remaining core group were in agreement. Therefore, we also computed precision and t-Test evaluations against the judgments of this core group alone. The results are presented in Table 9 and Table 10. Most importantly, KP still provides the best (in fact, further improved) results when the judgements provided by the core group is considered.

6. CONCLUSION AND FUTURE WORK

In many structured content, entries get their context implicitly from their neighbors, ancestors, as well as their descendants. In this paper, we first presented a technique to identify contextual relationships between entries through their relative contents. We then presented a relative content preserving keyword propagation technique to enrich entries in a given structure. The novel keyword weight propagation

technique enables keyword based indexing of hierarchical web data. The experiments showed that keyword propagation algorithm described in this paper provides a significant improvement in results and captures the domain context better than alternative solutions. Future work includes evaluation and incorporation of other types of semantic cues (such as [23, 21, 24, 18, 16]) for more informed keyword propagation.

7. REFERENCES

[1] Yahoo Directory. http://dir.yahoo.com/.

[2] Google Directory. http://directory.google.com/.

[3] A. Balmin, V. Hristidis, and Y. Papakonstantinou. ObjectRank: Authority-based keyword search in databases. In *Proceeding of the international conference on Very large data bases*, 2004.

[4] G. Bhalotia, A. Hulgeri, C. Nakhe, and S. Chakrabarti. Keyword Searching and Browsing in Databases using BANKS. In *International Conference on Data Engineering*, 2002.

[5] K. Bharat and M. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of the ACM SIGIR Conference*, 1998.

[6] S. Brin and L. Page. The anatomy of a large scale hypertextual web search engine. In *International World Wide Web Conference*, 1998.

[7] K.S. Candan, and W.S. Li. Using Random Walks for Mining Web Document Associations. *PAKDD*, 2000.

[8] K.S. Candan, and W.S. Li. Reasoning for Web Document Associations and its Applications in Site Map Construction. *Journal of Data and Knowledge Engineering*, 43(2), 121-150, 2002.

[9] S. Chakrabarti, M. Josdhi, and V. Tawde. Enhanced Topic Distillation using Text, Markup Tag, and Hyperlink. In *Proceedings of the ACM SIGIR Conference*, 2001.

[10] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv. XSEarch: A semantic search engine for XML. In *Proceeding of the international conference on Very large data bases*, 2003.

[11] J. Dean and M. Henzinger. Finding related pages in the World Wide Web. In *International World Wide Web Conference*, 1999.

[12] S. Dumais, and H. Chen. Hierarchical classification of Web content. In *SIGIR*, 2000.

[13] J. Faraway. Practical Regression and Anova using R. http://www.stat.lsa.umich.edu/~faraway/book.

[14] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram. XRANK: Ranked keyword search over XML documents. In *Proceedings of the ACM SIGMOD Conference*, 2003.

[15] V. Kacholia, S. Pandit, S. Chakrabarti, and S. Sudarshan. Bidirectional Expansion For Keyword Search on Graph Databases. In *VLDB*, 2005.

[16] J.J. Jiang, and D.W. Conrath. Semantic Similarity Based on Corpus Statistics and Lexical. In *Proceedings of the International Conference on Research in Computational Linguistics*, 1997.

[17] J.W. Kim, K.S. Candan, and M.E. Donderler. Topic segmentation of message hierarchies for indexing and navigation support. In *WWW*, 2005.

[18] D. Lin. An Information-Theoretic Definition of Similarity. In *International Conference on Machine Learning*, 1998.

[19] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46:604–632, 1999.

[20] T. Qin, T. Liu, X. Zhang, Z. Chen, and W. Ma. A Study of Relevance Propagation for Web Search. In *SIGIR*, 2005.

[21] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and Application of a Metric on Semantic Nets. *IEEE Transactions on Systems, Man and Cybernetics*, pp. 17-30, 1989.

[22] D. Rafiei and A. Mendelzon. What is this page known for? Computing web page reputations. In *WWW*, 2000.

[23] P. Resnik. Semantic Similarity in a Taxonomy: An Information-Based Measure and Its Application to Problems of Ambiguity in Natural Language. *Journal of Artificial Intelligence Research*, vol. 11, pp. 95-130, 1999.

[24] R. Richardson, A.F. Smeaton, and J. Murphy. Using WordNet as a Knowledge Base for Measuring Semantic Similarity. Dublin City Univ., 1994.

[25] G. Salton, E.A. Fox, and H. Wu. Extended Boolean information retrieval. *Communications of the ACM*, 26(11):1022-1036, Nov. 1983.

[26] M. Sanderson, and J. Zobel. Information Retrieval System Evaluation: Effort, Sensitivity, and Reliability. In *SIGIR*, 2005

[27] J. Savoy, G. Xue, and W. Ma. Ranking Schemes in Hybrid Boolean Systems: A New Approach. *Journal of the American Society for Information Science*, Vol. 49, Issue 3, pp. 223-253, 1997.

[28] C. Shah, and W. Bruce Croft. Evaluting High Accuracy Retrieval Techniques. In *SIGIR*, 2004.

[29] A. Shakery, and C. Zhai. Relevance Propagation for Topic Distillation UIUC TREC-2003 Web Track Experiments. In *Text REtrieval Conferences*, 2003.

[30] R. Song, J.R. Wen, S. Shi, G. Xin, T.Y. Liu, T. Qin, X. Zheng, J. Zhang, G. Xue, and W.Y. Ma. Microsoft Research Asia at Web Track and Terabyte Track of TREC 2004. In *Text REtrieval Conferences*, 2004.

[31] W. Wibowo, and H.E. Williams. Strategies for minimising errors in hierarchical web categorisation. In *CIKM*, 2002.

[32] Z. Wu, and M. Palmer. Verb Semantics and Lexical Selection. In *Annual Meeting of the Association for Computational Linguistics*, 1994.