Approaches to Text Retrieval for Structured Documents

Gerard Salton and Chris Buckley*

January 11, 1990

Abstract

Documents such as textbooks, dictionaries, and encyclopedias are inherently structured, in the sense that they are meant to be used selectively by skipping from section to section instead of reading sequentially from one end to the other. Experiments are described designed to provide selective reading lists for textbook materials in answer to questions submitted by the user population. A textbook in information science is used for experimental purposes.

1 Structured Text Collections

It is well known that collections of written text are inherently structured. For example, explicit text relationship indicators are often provided in the form of cross-references, footnotes and citations, and implicit content relationships exist between the sentences in a given paragraph, and between different paragraphs, and different documents.

Such text relationships have been used in the past in various ways — for example, in collection clustering, and relevance feedback. Clustering techniques are designed to group documents into affinity classes, making it possible to carry out efficient collection searches and to retrieve classes of similar items in a single search operation [1-4]. Analogously, relevance feedback is used to improve search statements, and hence to retrieve new relevant items, by utilizing relevance assessments obtained from system users for previously retrieved documents [5-7].

The recent work in the *hypertext* area also uses text structure to simplify text traversal and text retrieval operations [8-10]. In that case, links are placed

*Department of Computer Science, Cornell University, Ithaca, NY 14853. This study was supported in part by the National Science Foundation under grant IRI 87-02735.

between related pieces of text, and these links are followed during retrieval to identify related texts, or text excerpts. Structured text representations are especially useful for texts that are not meant to be read sequentially from one end to the other. The links are then used to provide selective retrieval of certain linked text portions.

In an information retrieval context, several text-structuring problems must be faced:

- How to subdivide the texts into linking units that provide advantages in information retrieval.
- How to identify relatable text portions while supplying the corresponding text links.
- How to traverse a linked text for retrieval purposes.
- How to measure the retrieval effectiveness in a structured text environment, compared with the retrieval in ordinary text collections.

These questions are examined in the remainder of this study.

2 Automatic Text Structuring

In some environments, the basic text structuring task is largely self-defined. For example, when dictionaries, thesauruses, or encyclopedias are used as a retrieval base, the linking unit almost surely consists of individual entries or encyclopedia units. In that case the library system is designed to facilitate jumping from one dictionary entry or one encyclopedia article to a related one.

When more general texts are processed, appropriate linking units might be defined before structured text searches are possible. In the search operations implemented for the complete texts of Shakespeare on the NeXT machine, the basic retrieval unit was chosen either as one complete Shakespeare sonnet (14 lines), or else one scene of a Shakespeare play — typically 100 to 200 lines of text. These individual text units constitute *local documents* that are treated as separate text units within the context of the larger document collection.

When conventional running texts are available in a retrieval environment, such as complete books or journal articles, a whole book chapter may cover many pages of text and deal with a variety of different topics. An individual text sentence, on the other hand, is often very confined and difficult to interpret out-of-context. In the experiments conducted in this study, complete *paragraphs* of text are used as linking units, the assumption being that the content within a paragraph is sufficiently homogeneous to be used as a basic unit for retrieval purposes.

When text paragraphs are treated as retrieval units, the content linking task can then be handled in the following way:

a) Individual text paragraphs are recognized.

b) An indexing system is used to identify paragraph content and to assign content terms.

c) The paragraph descriptions are compared and links are supplied between paragraphs with sufficiently high content similarily.

The important step is the paragraph content identification. Over the last few decades, viable *automatic indexing* systems have been developed that are capable of assigning to each text item a set of important terms used for content identification. Given a text item D_i (a particular text paragraph), the text content is often represented as a set, or vector, of terms $D_i = (d_{i1}, d_{i2}, ..., d_{it})$ where d_{ik} represents an importance factor, or *weight*, of term T_k assigned to item D_i [6, 7, 11, 12].

A high performance term weighting system normally takes into account the frequency with which a term is used in a particular document, the number of documents in a collection to which a term is assigned, and the document length or number of terms occurring in a document. The so-called *tf* $\times$ *idf* (term frequency times inverse document frequency) strategy assigns high term weights to text elements that occur frequently inside a particular document but relatively rarely in the collection as a whole. Terms with a large *tf* $\times$ *idf* factors are know to be important for content indentification purposes. [13-15]

Given two paragraphs D_i and D_j both represented by term vectors, a similarity measure may be computed between the two items based on the number and the weight of jointly assigned terms. Mathematically, the similarity between two text items $D_i = (d_{i1}, d_{i2}, ..., d_{it})$ and $D_j = (d_{j1}, d_{j2}, ..., d_{jt})$ can be measured by the inner product between the corresponding term vectors as follows:

$$Sim(D_i, D_j) = \sum_{k=1}^{t} d_{ik} \cdot d_{jk} \qquad (1)$$

where t is the total number of assignable content terms. When *tf* $\times$ *idf* weights are used to reflect term importance and the similarity computations are normalized for document length, the pairwise similarity Sim (D_i, D_j) produces values between 0 and 1.

Global similarity computations such as those of expression (1) are usable for document classification when classes of items are defined consisting of items exhibiting a sufficiently high pairwise global similarity. For text classification purposes pairwise similarity measures must be computed between paragraph pairs and grouping criteria must be defined to generate classes of mutually related text items. Typically, hierarchical text classification systems can be constructed by first forming small groups of highly related items (where each group consists of a small number of items with a large pairwise similarity). The small tighly related classes may be expanded into larger groupings with a smaller overall similarity. When this process is continued, one large heterogeneous class is formed at the end consisting of all items in the text collection. [1-4]

Fig. 1 shows an excerpt of a cluster (class) hierarchy constructed for the paragraphs of a textbook in information science.[16] In the illustration of Fig. 1, three low-level clusters are defined consisting of items (397 and 791), (642 and 644), and (655 and 656). The global similarity coefficients obtained for the respective term vectors (see expression (1)) are included in the figure ranging from 0.605 for items 642-644 to 0.556 for 397-791. The two groups consisting of (642, 644) and (655, 656) are themselves grouped into a larger class with an overall similarity of 0.410, implying that the smallest similarity between any pair in the group (642, 644, 656, and 656) is 0.410. Finally the group of 4 items is joined with another group of two items consisting of (397, 791), the global similarity of the complete set of 6 items being 0.329 (the smallest similarity between some element in group (397, 791) and some other element in (642, 646, 655, 656) is 0.329.

A hierarchical representation of the cluster of Fig. 1(a) is shown in Fig. 1(b), where the actual documents (paragraphs) are represented by the leaves of the tree, and the interior nodes specify the respective clustering similarity. In the illustration of Fig. 1(b), the four paragraphs of chapter 9 of [16] cover topics dealing with the generation of word stems, the assignment of content identifiers to the documents of a collection, and the general automatic indexing process. Item 397 from chapter 7 discusses text decomposition of words and affixes, and item 791 from chapter 11 deals with word morphology from a linguistic viewpoint. All of these topics are related to word stemming and automatic indexing.

In principle, a hierarchival document or paragraph classification can be used directly to define an appropriate linking structure usable for text retrieval:

a) An incoming query may be compared with all existing paragraph descriptions.

b) The best matching text paragraphs can be retrieved (say paragraph 642 in the illustration of Fig. 1).

c) Additional paragraphs are retrieved from the same cluster, assuming that the reader wishes to see more output materials.

d) The search may be expanded to adjacent clusters of items (say 655, 656), if the user wishes to obtain still more information.

When the paragraph similarities are sufficiently high – say above 0.400 on a similarity scale ranging from 0 to 1 – properly related paragraph sets may emerge with such a cluster search process.

In practice, as the clustering example of Fig. 1 shows, jumping from cluster to adjacent cluster often uses links of low similarity, possibly indluding large-scale topic changes. A greater degree of confidence in the appropriateness of the global paragraph linking mechanism may be gained by constructing chains of mutually similar items, where item A is closely related to item B, which is in turn closely linked to C, and so on. An iterated similarity computation system may then be used to construct linked paragraph chains, starting with one or more seed items known to be relevant to the user:

a) Each seed item is compared with all other text items in the collection, and a similarity threshold is used to identify one or more related items.

b) The related items are used next as seed items and the search process is iterated to produce still more related items; the similarity threshold may be varied form one iteration to the next to control the number of related items retrieved in each iteration.

In the foregoing discussion, all paragraph comparisons are assumed to be carried out globally, by comparing the term vectors attached to the respective paragraphs using the model of equation (1). The likelihood of useful paragraph links may be increased in some circumstances by comparing *sentences* in highly matching paragraph pairs, and retrieving a linked item only if the global similarity with a seed item exceeds a stated treshold, *and* if at least one (or more) matching sentence pairs are found in the respective paragraph pair. Substantial evidence exists that the presence of highly matching sentences in pairs of paragraphs provides evidence of content relationship between text excerpts. A pairwise sentence comparison may then be performed optionally for paragraphs with high global similarity, in addition to the global paragraph comparisons.

When sentences are compared, a global similarity based on normalized term weights, such as those of expression (1), may not be suitable. For short sentences involving only one or two significant terms, the global similarity with normalized weights will produce perfect similarity coefficients of 1 for many sentence pairs. Furthermore, the inverse document frequency (*idf*) factor that depends on the number of documents in which a term occurs is not unambiguously defined in a sentence context.

For sentence similarity computations, it then appears preferable to use as a weight for term k in sentence S_i, the term frequency tf_{ik}, representing the number of occurences of term k in S_i. In addition, an extra weight might be attached to matching sequences of significant terms that occur adjacently in the respective sentences, or that occur in close proximity of each other in the sentences. For purposes of this study, the similarity between sentences S_i and S_j is obtained simply as the sum of the minimum term frequency weights of matching terms in the sentences S_i and S_j:

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.