A Comparison of Indexing Techniques for Japanese Text Retrieval

Hideo Fujii & W. Bruce Croft

Computer Science Technical Report 93-35
April 1993

A Comparison of Indexing Techniques for Japanese Text Retrieval

Hideo Fujii* and W. Bruce Croft**
Computer Science Department
University of Massachusetts, Amherst, MA 01003

e-mail: * fujii@cs.umass.edu ** croft@cs.umass.edu

Abstract:
A series of Japanese full-text retrieval experiments were conducted using an inference network document retrieval model. The retrieval performance of two major indexing methods, character-based and word-based, were evaluated. Using structured queries, the character-based indexing performed retrieval as well as, or slightly better, than the word-based system. This result has practical significance since the character-based indexing speed is considerably faster than the traditional word-based indexing. All the queries in this experiment were automatically formulated from natural language input.

1. Introduction

There are two major difficulties in applying text retrieval techniques already developed for English (or other European languages) to Japanese. The difficulties are: (1) the intensive use of the *Kanji* (=Chinese) character set which is an ideographic system, and has 2000-3000 characters in practical use [INT+86]; and (2) as an agglutinating language, Japanese sentences have no spaces between words.

In previous Japanese full-text IR studies, there are few experimental results using probabilistic or ranking approaches. Most studies use Boolean exact matching approaches, although the limitation of the exact-matching approach is recognized. For example, Negishi [Neg89] discusses the low precision of such retrieval approaches, and his expectation of improvements from the adjacency operator. Probabilistic retrieval has not been thought of as a practical method in Japan [Sas88]. Even in recent review papers about Japanese full text IR, there is no discussion about probabilistic ranking ([Neg92], [OKT92]). On the other hand, there are many studies focusing on the Kanji character's semantic discrimination ability ([GHO+84], [Sas89], [MUH+91], [HGH+89]), but no experimental results of the application of character level indexing for IR.

A number of studies have shown that good retrieval performance can be achieved by probabilistic methods such as the inference network model [Tur91], [TC91], but we know little about how Japanese language features affect performance in a total system. The advantage of the inference net model for Japanese retrieval is that queries can be expressed as structured combinations of characters.

In this paper, we report evaluations of inference network-based retrieval using the major Japanese text indexing techniques - *character-based* (CB) which treats every character as an index term, and *word-based* (WB) indexing which uses each word (or word stem) as an index term.

This paper has three research contributions. First, we get performance data for probabilistic retrieval for Japanese. Second, the data shows that character-based indexing performs retrieval as well as a word-based system. Third, this retrieval performance can be achieved by automatic structured query formulation from the natural language input. The second point has a practical significance since the character-based indexing speed is considerably faster than word-based indexing.

2. The INQUERY/JINQUERY System

2.1 Japanese INQUERY System, JINQUERY

For our experiments, we implemented a Japanese IR system, JINQUERY, based on an inference network retrieval system, INQUERY [CCH92]. Since the inference network retrieval is an extension of the probabilistic retrieval model, INQUERY is language independent. Using an INQUERY retrieval engine, JINQUERY provides the Japanese language oriented modules.

JINQUERY consists of two modules. One is an indexing module to create a database from source texts. Since the Japanese words in a text are not separated by spaces, a *segmentation* program is used

to identify and extract index terms from the documents. In this experiment, we used the JUMAN segmentation program [MKM+91] for word-based indexing.

The other is a query front-end module to formulate a structured query using Japanese grammatical aspects. When a query is input as a natural language, JUMAN is again used for the query segmentation.

JINQUERY was implemented on a SUN-IPC Unix workstation using the C language.

2.2 Inference Network Retrieval Model

The Bayesian probabilistic inference network [Pea89] is specialized for IR as the *document retrieval inference network model.* The details of this model are discussed in [Tur91] and [TC91]. A simple document retrieval inference network is shown in Figure 1. The network is represented as a directed acyclic graph in which each node corresponds to the propositional variable (*true* or *false*) and the arc between two nodes represents the dependency between them. The dependency is defined as a conditional probability for the relevance, which is interpreted as the *belief* in the course of inference.

Rretrieval is viewed as an estimation of a conditional probability to satisfy an *information need* (I) given the document, i.e., P(I | Document). An information need is represented by a query node (q) at the root of the network. Between the query (=root) and documents (=leaves), there are two subnetworks - a *document network* and a *query network.*

The document network consists of *document nodes* (d_i's) and *content representation nodes* (r_j's). Every document node corresponds to the existence of each document. The content representation nodes represent concepts for the documents. When a concept is observed in a document, an arc links them.

A query network is built in the retrieval session. The session may contain the process of (automatic or manual) query formulation, and/or relevance feedback. A *query* (q) is analyzed into several *query concepts* (c_i), and query concepts are attached to the document network in the query processing. Between query concepts and the query node, it is possible to construct a set of intermediate node levels to express a complex query structure constructed by various operators. This structure has a corresponding *canonical link matrix* representation.

By the attachment of the document network and a query network, the system can compute the probability of the desired documents, then it can rank them according to these estimations.

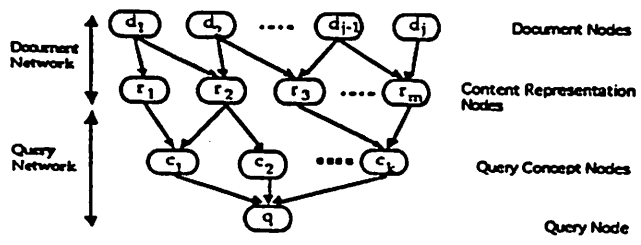

Figure 1. A simple inference network for document retrieval.

3. Indexing Techniques

3.1 General Issues

Indexing is the use of language to describe the documents and user's information needs. Index terms are derived from the document text or the user's input. Indexing is done by either human experts or an *automatic indexing* program. *Manual document indexing* is labor-intensive and time-consuming work, and has the drawbacks of lack of uniformity and indexer-user mismatch. In contrast, automatic indexing has the advantage of bias-free uniformity and efficiency. Furthermore, studies show that automatic indexing has approximately the same retrieval performance as manual indexing [Sal86] - at least in English. In Japanese also, there are many studies about the effectiveness of various automatic indexing techniques (e.g., [TAK+81], [Kim87], [Kam89], [Tom89], [ISN92]).

Two factors affect retrieval performance. One is the *exhaustivity*, which refers to how much the index covers the topics of the document; the other is the *specificity*, which refers to how precise the index is. Since these factors have opposite effects on recall/precision retrieval performance, we need to control from both sides [Van79].

A word, as a whole, has more specific meaning than its Kanji constituents. Kanji constituents in a word are even not necessary to be exhaustive. Under these conditions, CB indexing needs *post-coordinating* (i.e., run-time) handling for better retrieval performance.

Automatic indexing usually proceeds by the following steps: (i) get words in each document, (ii) exclude stopwords from them, (iii) do stemming to produce index terms, (iv) compute the pre-coodination information (e.g., term frequency etc.) and pointers from the term to documents to build an inverted file. Our JINQUERY system works in a similar way.

See the detailed discussion about indexing in e.g., [SM83], [Fra92], and [CCH92] for indexing in an inference network system .

3.2 Indexing of Japanese Texts

3.2.1 Basic Indexing Techniques for Japanese

There are four basic indexing approaches: (1) subcharacter-based, (2) character-based (CB), (3) n-gram and (4) word-based (WB).

The majority of the Kanji characters are not minimum semantic units in a word. These characters are composed by parts which are called *bushu*. *Subcharacter-based* indexing extracts them as index terms. The problem here is that some bushu are like meaningless strokes, and some others are very uncommon. The semantics, therefore, are often too vague. For this reason, we did not evaluate the subcharacter-based method in this experiment.

The *character-based* method indexes each character in the document. N-gram indexing uses (overlapping) n-characters in words for indexing. That is, the CB method is equivalent to 1-gram indexing.

The simple n-gram method will not be suitable for Japanese IR, since it ignores the important language structure of written Japanese. Heterogeneous character classes are used, where each class has a clear linguistic functional role: Kanji terms are primarily used to express abstract or complex important concepts ideographically; Katakana words are mostly phonetic loan words especially from English; Hiragana characters are used for inflection or other functional words such as particles, auxiliary verbs, etc. Furthermore, one can often see English alphabetic words in a Japanese text, especially as proper nouns such as "IBM" or "C" [San86].

Because of the above facts, we used the following method in our character-based indexing experiment: All Hiragana's are dropped from the text; each Kanji is individually indexed; sequences of Katakana characters or English characters are extracted as index terms. As we see in the next section, this algorithm is very fast since the program needs just to distinguish the class of the characters. (In the following discussion, we call this approach "CB".)

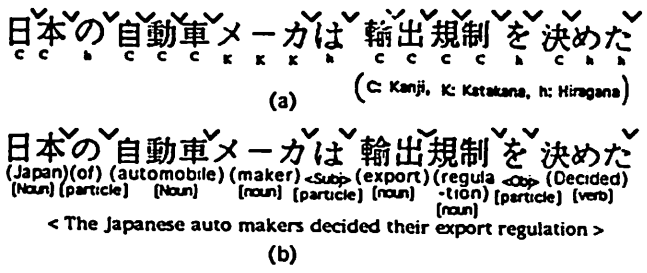

Figure 2. An example of character-based (a) and word-based (b) term boundaries.

The word-based technique extracts words (or some normalized forms like word stems), as the index terms. Although this is a standard approach for Japanese automatic indexing (e.g., [KMK80]), the problem of Japanese is how to identify "words" (*morphemes*) in a character sequence which is not delimited by blanks. This identification process is called *segmentation*. The segmentation problem is discussed in Section 3.2.3.

Examples of Character-based and word-based indexing are shown in Figure 2 (a) and (b). Figure 3 and 4 show how these indexing methods fit in the inference network framework.

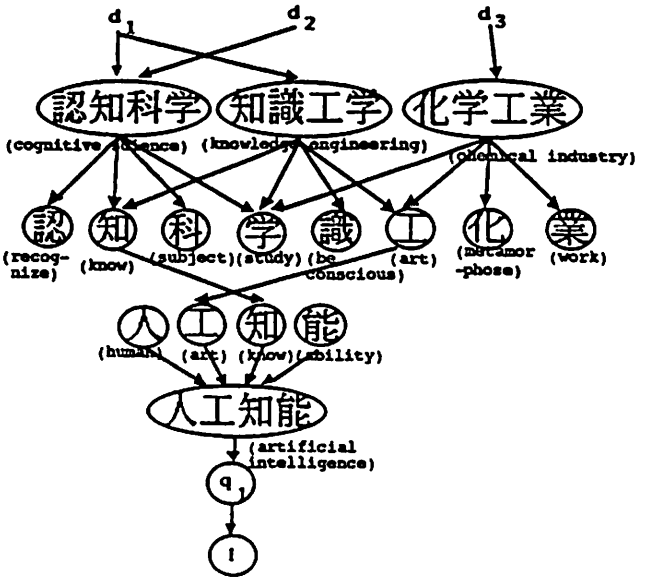

Figure 3. An example of Japanese character-based inference network.

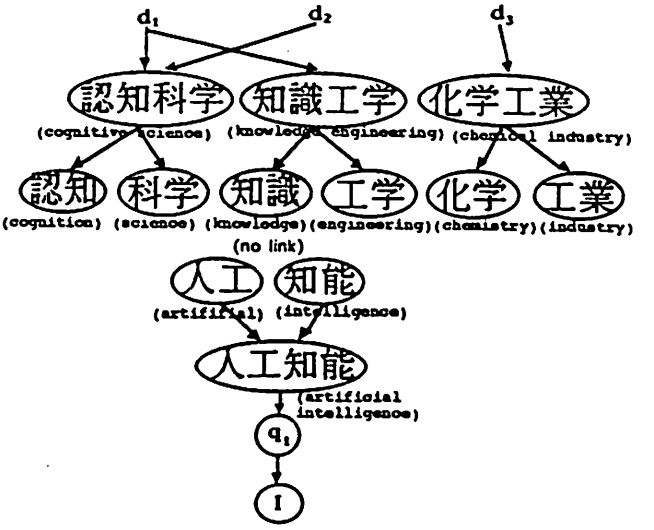

Figure 4. An example of Japanese word-based inference network.

In Figure 3, both document words and query words are decomposed into a set of Kanji characters. In Figure 4, each Kanji compound noun is divided into its components.

In CB indexing, the connections between a query word and the query characters are formulated by structured query operators. In WB indexing, the connections between a query compound word and the compound elements are structured. The operators used to define the structure are part of the inference net model.

3.2.2 Problems of Word Compounds

As in German, Kanji words are frequently composed into a long compound word (Figure 5). Each compound element is often referred as a *short unit keyword*, and the compound as a *long unit keyword* in Japanese IR literature. Each method has advantages and disadvantages as index terms. Someya [Som85] summarized that long keywords are suitable for Boolean retrieval, and short unit keywords are suitable for contextual retrieval using *adjacent* operators. Short unit keywords demonstrated better retrieval performance [KHK88]. We also took the short unit keyword approach for WB indexing.

労働省公共企業体等労働委員会委員長
(labor) (ministry) (public) (corporation) (bodies) (etc.) (labor) (committee) (meeting) (committee) (head)

<Ministry of Labor, Public Corporation and National Enterprise Labor Relations Commission, Chairperson>

Figure 5. An example of a Japanese long unit keyword.

3.2.3 Segmentation Problems

We have seen how the CB indexing system separates each Kanji character and Katakana word in the text. On the other hand, there are three levels of algorithms for the WB indexing : 1) character class segmentation, 2) closed lexicon segmentation, and 3) open lexicon segmentation.

The *character class segmentation* drops all Hiragana out from the text, then the rest are extracted as index terms. This method works as efficiently as CB, but suffers from low accuracy in finding the word boundaries.

The second segmentation approach is *closed lexicon segmentation*. It improves the word recognition by using inflectional or derivational lexicons, and the computational cost doesn't increase much since this lexicon would be comparatively small. This method doesn't, however, solve the problem of Kanji compounds discussed in the previous section.

The last approach, *open lexicon segmentation* , is aimed at solving the Kanji compound problem, discussed in the previous section. This is very important for Japanese WB indexing, since important concepts in the text are very frequently represented in Kanji compounds. Like the Japanese Kana-Kanji-transformation algorithms, this method usually uses a large dictionary look-up for the heuristic algorithm such as the longest string matching, and checking word class connectivities [Tan89]. The problems with these methods are their expensive computational cost, and the difficulty of building and maintaining a sufficiently large term dictionary, especially for proper nouns.

As we will see later, we evaluate CB indexing, and WB indexing by an open lexicon segmentation program, JUMAN.

In a few cases of CB retrieval, *segmentation mismatch* between some query terms and document terms were observed. For example, a Katakana query word *WâkuSutêshon* (=workstation) was segmented into *Wâku* and *Sutêshon* by JUMAN for query segmentation, but the database was indexed by character classes in CB indexing. So, the word is stored as a single string *WâkuSutêshon*. To correct this problem, we will need post-processing after the segmentation.

3.2.4 Synonymy and Multi-Spellings

In addition to common synonyms, Japanese uses synonyms of loan words from foreign countries. For example, to express *running* in English, there are at least three variations: (i) *Hashiru-koto*, this is an example of original Japanese usage, (ii) *SouKou*, this Kanji term is a word from Chinese, and (iii) *ran'ningu*, this is a phonetic Katakana translation from "running". To avoid this problem, it is necessary to build a thesaurus, but it will be large. Furthermore, each of these words has often different meaning or role in a context. So, we did not address this problem in this experiment.

1. Abbreviations

神戸製鋼＝神鋼 (Kobe-Steel Co.)

電子計算機＝電算機 (computer)

2. Partial Matching

旅行≈旅行社≈海外旅行≈「旅に行く」
(travel) (travel agency) (travel abroad) ("take a trip")

保険≈生命保険≈保険料
(insurance) (life-insurance) (insurance cost)

3. Synonyms/Related Words

映画≈映像≈東映
(movie) (visual image) (Touei : [a movie company name])

自動車≈乗用車≈四輪車
(automobile) (passenger car) (4-wheel vehicle)

Figure 6. Examples of thesaurus effects.

The CB indexing has an advantage for *thesaurus effects*. Since the Kanji is an ideogram, if two words share a Kanji character, some shared conceptual elements are observed between them. Word stemming works as a recall enhancement device, but this CB technique has a wider range of effects. Some examples are shown in Figure 6.

We see multi-spelling problems sometimes in English, e.g., "color" and "colour", but in Japanese, they occur much more frequently. To identify these equivalences, we need various normalization processes [HKF89]. The following are some cases [INT+86]:

(A) *Okuri-gana* (= inflectional Hiragana) - This is often dropped from a verbal expression such as verb plus noun (e.g., *Yo(mi)-Mono* = reading material), or verb plus verb (e.g., *Yo(mi)-kiru* = finish reading). Nakamura [Nak87] proposed a normalization algorithm for this problem. This problem doesn't occur in CB indexing since it drops all Hiragana.

(B) Katakana words - Most Katakana words are phonetic translations from English words, and there is no standard. For example, a word *interface* has five different Katakana spellings, *intâfêsu*, *intâfeisu*, *intafêsu*, *intafeisu* and *intafeusu* [TOT83].

(C) Multi Character Classes - Since Japanese uses phonetic Hiragana and Katakana, Kanji words can be also spelled in them.

(D) Multiple glyphs - Sometimes a single Kanji has different character forms (*glyphs*), for example, the old form and modern form of a character. *Character unification* is necessary to avoid this problem.

3.2.5 Stemming Problems and Morphological Changes

In English indexing, *derivational* or *inflectional* ending of words are stemmed out to identify equivalence class. For the derivational changes, a segmentation program like JUMAN usually recognizes the prefixes or postfixes in a word.

In JINQUERY, inflected forms of verbs (and adjectives) are converted into the root form as their index term using the function of JUMAN. Nouns don't have any inflectional changes (e.g., number, gender, etc.) at all. This is good news for IR, since the Japanese heavily use verbal Kanji/Katakana nouns with *Sahen-verbs* (i.e. *suru* = "do" in English, e.g., *puroguramu* (n.)+*suru*=to "program"), and they would be integrated with common nouns. Similar problems occur for some adjectives(*Na-adjective*, e.g., *kon'nan* ("difficulty"(n.))+*na* ="difficult"(adj.)), but this cannot be solved in the segmentation stage, since the inflectional part of a word cannot be separated from the root [Yam92]. To solve this problem, some post processing after segmentation is necessary. We are working on this improvement. Note that this difficulty is raised only in a WB system, not in a CB system.

There is another problem that many Japanese verbs have similar, but different forms of *transitive* and *intransitive verbs*. For example, *Ugoku* is an intransitive verb ("I *move* to a new house."), but *Ugokasu* is transitive ("I *move* the book on the table."). Treating them as a single representation in WB indexing, string normalization is necessary as seen for various multi-spelling problems. But again, this is not a problem in CB indexing since it drops all Hiragana inflectional endings. In this sense, the CB index is more robust than WB.

3.2.6 Stopwords and Word Categories

In English IR systems, terms in a document are usually classified into index terms and stopwords. This selection is usually done by matching between a document term and a *stopword list*. A stopword list contains very frequent (mostly functional) words such as articles, pronouns, etc. On the other hand, most Japanese IR systems extract only nouns as retrieval keywords.

JINQUERY takes a different approach. The JUMAN program not only segments the words, it also outputs their lexical categories and subcategories. Given this, we throw out words in functional categories such as particles, Sa-hen verb (*suru*), etc. Other categories such as nouns, verbs (exclude Sa-hen verbs), adjectives, prefixes/postfixes, undefined terms (which JUMAN couldn't recognize) etc. are used as the index terms.

4. Japanese Query Formulation

In English, using statistical phrases in a *structured query* with retrieval operators is an effective method for improving retrieval performance ([Fag87], [CTL91]). We did experiments with the same technique in Japanese.

As a pretest before retrieval experiments, we compared automatically formulated queries and manually crafted queries over our test queries. The result was that more than 90% of queries were identical in any formulation method. We decided to evaluate the automatically formulated queries.

In our experiments, we examined four models: 1) NLQ, 2) SHORT, 3) LONG, and 4) JOINED. Each model is described in the following sections, and Figure 7 is a sample input query which is commonly used for them.

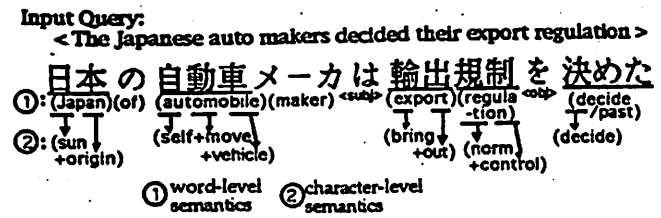

Figure 7. A sample input query and the semantic constituents.

4.1 Natural Language Query

The *Natural Language Query* (NLQ) does not assume any structure within the query. A natural language input query is transformed into a form :

q_{NLQ} = #SUM($t_1, t_2, \ldots, t_n$);

Here, #sum is an operator to evaluate the node as an average of child node beliefs in the inference net, and t_i is an index term of the query [Tur91]. Index terms are selected from input query only when it is not a stopword category word (e.g., particle, auxiliary verb, sahen-verb, etc.).

In CB indexing, terms constructed only by Hiragana are not selected as well. This term selection rule is applied in any following formulation, too.

Example:

[CB] #SUM(<sun><origin><self><move><vehicle> <maker><bring><out><norm><control> <decide>);

[WB] #SUM(<Japan><automobile><maker> <export> <regulation> <decide>);

4.2 SHORT Query

The *SHORT* query is an attempt to capture a "short unit keyword" structure which roughly corresponds to the word in English. This is only the case for CB indexing, since in the WB system, the short unit keywords are given as primitives of indexing language. SHORT query has a form :

q_{SHORT} = #SUM($T_1, \ldots, T_n$);

T_i := #op($t_{i1}, t_{i2}, \ldots, t_{im}$) , and

$t_{i1}, \ldots, t_{im}$ forms a short unit keyword.

Here, #op is an operator to tie the operands, and we examined the *phrase* operator (#phrase, window size = 1 and 3) operator and *proximity* operators (#1 and #3) ([Fag87], [CTL91], [CCH92]). (Following sections as well.)

Example:

[CB] #SUM(#phrase(<sun> <origin>) #phrase(<self><move><vehicle>) <maker> #phrase(<bring> <out>) #phrase(<norm> <control>) <decide>);

4.3 LONG Query

The *LONG* query is an attempt to capture a structure of long unit keyword, i.e., compound nouns. Intuitively, this structure similarly corresponds to the noun phrase in English. A compound noun in a LONG query is defined by a regular expression. JINQUERY produces the LONG query as a structured query such that :

q_{LONG} = #SUM($T_1, \ldots, T_n$);

T_i := #op($t_{i1}, t_{i2}, \ldots, t_{im}$) , if $t_{i1}, \ldots, t_{im}$ forms a compound noun(NC),

:= t_i , otherwise.

NC := [<prefix>* <noun>+ <postfix>*]+ .

Example:

[CB] #SUM(#phrase(<sun> <origin>) #phrase(<self> <move> <vehicle> <maker>) #phrase(<bring> <out> <norm> <control>) <decide>);

[WB] #SUM(<Japan> #phrase(<automobile><maker>) #phrase(<export> <regulation>)<decide>);

4.4 JOINED Query

The *JOINED* query is an attempt to capture a structure of a maximal sequence of noun phrases which are connected by a particle *"no"* (= "of" in English) or its variants. This structure is a model based on a fact : such particles are often dropped from the sentence, and produces a new long noun compound. JOINED structure is defined by a regular expression. The form of JOINED query is :

q_{JOINED} = #SUM($T_1, \ldots, T_n$);

T_i := #op($t_{i1}, t_{i2}, \ldots, t_{im}$) , if $t_{i1}, \ldots, t_{im}$ forms a JOINED structure(JS).

:= t_i , otherwise.

JS := NC2(NV NC2)*

NC2 := <adjective/adnoun>*[<prefix>* <noun>+ <postfix>*]+

NV := *"no"* | <noun-connect-particle> | <case particle> *"no"* | <noun-connect-particle> *"no"*.

Example:

[CB] #SUM(#phrase(<sun> <origin> <self> <move> <vehicle> <maker>) #phrase(<bring> <out> <norm> <control>) <decide>);

[WB] #SUM(#phrase(<Japan><automobile> <maker>) #phrase(<export> <regulation>) <decide>);

There is an imperfect case of the JOINED regular expression in parallel form, "A *of* B *and* C" (in English order), e.g., "king of the world and the queen" and "king of the earth and ocean". The current system produces #SUM(#phrase(king world) queen) and #SUM(#phrase(king earth) ocean), respectively. We need a natural language disambiguation technique to solve this problem.

4.5 Some Grammar Related Problems

Since this is an important area of natural language processing, we just mention the S-O-V word order problem. Although most of the words in a Japanese sentence are ordered quite freely, the verb is almost always placed at the end of the sentence. So, a typical word order format of the Japanese sentence is S-O-V (as is the majority of world languages [Tsu91]). Thus, the distance between subject and verb will be larger, but subject and object will be closer than in English. The effect of phrase or proximity operator for Japanese will be different from English.

5. The Experiments

5.1 Collections

The characteristics of the test collection often affect IR performance seriously. To compare the retrieval performance with other data, using a standard collection is very important for the experiment, but unfortunately, a standard test collection in Japanese, such as CACM collection in English, is not established yet.

So, we used a test collection which has 1101 newspaper articles on Japanese business and economics. It contains articles from three major Japanese newspapers, The Asahi, The Mainichi and Nihon-Keizai(Nikkei)-Shinbun. A larger collection is desirable for more reliable experiments. Table 1 is a statistical summary of this collection.

5.2 Test Queries

In this experiment, we used 30 test queries. Most queries are simple noun phrases like "joint ventures in the steel industry". Table 2 shows the summary. We are planning to try longer and more complex queries to see the difference.

5.3 Evaluation Methods

We evaluated the retrieval performance in terms of the precision value of the top-10 documents in the rank. We are currently working on creating a complete precision-recall table using full judgments of documents in the database.

5.4 Indexing Efficiency

In this section, we compare the indexing efficiency of CB and WB indexing using time and space measures.

(1) Time Efficiency

CB indexing was around 7 times faster than WB to parse and segment the source text. Sorting transactions and building an inverted file take around the same time in both cases. By simple extrapolation, WB indexing would take more than 2 weeks(!) for 1 GB data.

Table 1. Summary of Japanese test collection.

Collection Size	1,101 docs. (1,270 KB, 635 K Chars)	
Average Document Size (body)	500.0 chars (1.0 KB)	
Max Term Freq.	Char-Based 47 (*Sha* = "car")	Word-Based 24 ("IBM" and *ShouKen* = "stock")
Average Max_Term_Freq.	9.47	6.76
Term Categories	Kanji: 53.3% Katakana*: 5.0% Alphabets*: 1.0% Hiragana: 40.7% (Not-Indexed)	Indexed: Noun: 47.2% Verb: 5.9% Undefined: 11.1% Not-Indexed: 31.6%
Average Term Length (chars.)	Indexed: 1.33 Kanji: 1.00** Katakana: 4.62 Alphabets: 2.67	Indexed: 2.39 Noun: 2.06 Not-Indexed: 1.21
Unique-Term Categories (root form, Indexed)	Kanji: 28.3% (1746 Kanji) Katakana: 63.3% Alphabets: 8.4%	Noun: 38.4% Verb: 6.4% Undefined: 51.0% Others: 3.8%

(* Words are terms. ** Characters are terms.)

Table 2. Summary of test queries.

	30 queries 9.03 chars/query (max=17, min=4) 6.37 Kanji/query 1.17 Katakana/query 3.40 short unit keywords/query
SHORT	2.87 phrases/query 2.03 chars/phrase
LONG	2.20 phrases/query 3.15 chars/phrase
JOINED	1.07 phrases/query 7.17 chars/phrase

Table 3. Time efficiency of indexing methods.

	Character-Based	Word-Based
Segmentation + Parsing (rate)	4' 08" (1.0)	27' 44" (6.7)
Sorting + Build Inv.File	3' 53"	3' 46"
Total	8' 01"	31' 30"

(2) Space Efficiency

The total file size of CB and WB indexed file was 1.2 MB and 1.6 MB, respectively. A significant difference between them is that hash dictionary (to convert a term to its identification number) size by CB indexing was around half of WB's, since unique Kanji characters can quickly saturate. (Figure 8)

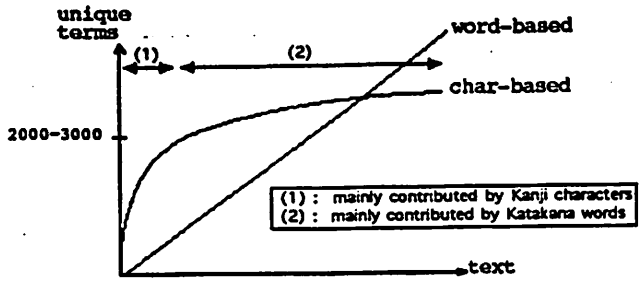

Figure 8. Increase of unique index terms.

The elimination of stopwords was around 40% in CB, and 30% for WB. In English, 50-70% is possible [Van79]. If we use a stopword list additionally, the efficiency may increase to the English level.

The average rate of (root form length)/(surface form length) was 99.2% for verbs, that is almost no impact on the space increasing or decreasing by our "stemming".

5.5 Retrieval Effectiveness

Table 5 shows the retrieval performance of CB indexing for every query model with various operators. Query models are ordered from simple to complex. Operators are ordered from less strict to more. Proximity operators work better for SHORT queries than for natural language queries (NLQ), but not for others probably because of its strictness.

In contrast, the phrase operator with the LONG model works significantly better than NLQ. Since SHORT and LONG correspond to the word and compound noun phrase, this result seems to be meaningful. JOINED queries don't work well for phrase operators. We might need a more complicated query structure for this model.

Table 4. Space efficiency of two indexing methods.

	Char-Based	Word-Based
Hash Dict. Size	220 KB (6173 entries)	423 KB (14847 entries)
Inverted File Size	989 KB (6,173 records) (160.2 b/rec)	1145 KB (14847 records) (77.1 b/rec)
Transaction Records	120,610	103,785
Indexed Terms	228,137	152,741
Unique Indexed Terms (root form)	6,173	14,847
Average Indexed Term Length	1.33 chars	2.39 chars
Dropped Text/Doc.Size	40.2% (Hiragana: 28.2%)	29.5%
Root Length/ Original Length (Average%)	-	Indexed: 99.8% Verb: 99.2%

Table 5. Character-based retrieval performance. (Top-10 precision), ( ): % changes from NLQ

NLQ	60.5

	SHORT	LONG	JOINED
#phrase (d=3)	57.9 (-4.3)	65.5 (+8.3)	60.8 (+0.5)
#phrase (d=1)	56.3 (-6.9)	66.6 (+10.1)	57.8 (-4.5)
Proximity (d=3)	62.9 (+4.0)	46.8 (-22.6)	16.0 (-73.6)
Proximity (d=1)	62.8 (+3.8)	48.3 (-20.2)	14.5 (-76.0)

d: window size

Table 6 shows the result of the retrieval performance of WB indexing. In this case, NLQ is almost as high as CB's best result. The phrase operator of JOINED form worked slightly better than NLQ, but not significantly. The results of the proximity operators are similar to the CB data, and they don't work well.

Table 6. Word-based retrieval performance. (Top-10 precision), ( ): % changes from NLQ

NLQ (=SHORT)	65.2

	LONG	JOINED
#phrase (d=3)	64.3 (-1.4)	65.5 (+0.5)
#phrase (d=1)	63.6 (-2.5)	66.3 (+1.2)
Proximity (d=3)	46.0 (-29.4)	10.2 (-84.3)
Proximity (d=1)	44.7 (-31.4)	10.7 (-83.6)

Table 7 is a performance comparison between CB and WB. Each data is the best value in all operator cases. The interesting result is that CB with comparatively simple LONG query structure was at least as good as any WB result.

Table 7. Retrieval performance of char-based & word-based indexing. (Top-10 precision, best results) ( ): % changes from char-based NLQ

	Character-Based	Word-Based
NLQ	60.5	65.2 (+7.8)
SHORT	62.9 (+4.0)	65.2 (+7.8)
LONG	66.6 (+10.1)	64.3 (+6.3)
JOINED	60.8 (+0.5)	66.3 (+9.6)

We can also make the observation that the retrieval performance appears to be good and certainly comparable with the performance of probabilistic systems with English.

6. Conclusion

In our experiments, we got the following results:

1) Using LONG model structured query with *phrase* operator, CB indexing achieved around the same or slightly higher retrieval performance than the best WB indexing result.

2) The above achievement was possible by automatic query formulation.

3) CB indexing is more efficient than WB in both time and space. CB indexing may have more chance to improve retrieval performance in the post-coordination stage. WB methods may get significant improvements at pre-coordination e.g., by using AI techniques for indexing, but more sophistication will increase the computing cost of document indexing.

We hope to get a clearer picture of the above research results in future work. We are going to overcome several limitations in this experiment such as the size of the test collection, incomplete relevance judgments, very simple test queries, and various indexing/segmentation problems, e.g., segmentation mismatch. Developing different models of structured queries for phrases will help to understand what is going on "behind the scenes" in Japanese IR.

Acknowledgments

This work was supported in part by the NSF Center for Intelligent Information Retrieval at the University of Massachusetts, Amherst. Jim Barnett and Ulf Hermjakob of MCC provided the version of the JUMAN segmenter that was used in these experiments and helped with the evaluation. Fumie Katsu also helped with the evaluation.

References

[CCH92] Callan J. P., Croft W. B., Harding S. M., "The INQUERY Retrieval System", 3rd International Conference on Database and Expert Systems Applications, 1992.

[CTL91] Croft W.B., Turtle H.R., Lewis D.D., "The Use of Phrases and Structured Queries in Information Retrieval", ACM SIGIR-91, pp. 32-45, 1991.

[Fag87] Fagan J., "Experiments in Automatic Phrase Indexing for Document Retrieval, A Comparison of Syntactic and Non-Syntactic Methods. Ph.D., Cornell University, 1987.

[Fra92] Frakes W.B., "Information Retrieval - Data Structures and Algorithms", edited by B. Yates, Prentice-Hall, NJ, 1992.

[GHO+84] Goto T., Hosono K., Oe T., et al., "The Relationship between Electrical Engineering and Chinese Characters Appearing in Articles Related to that Field", JICST 21st Annual Meeting on Information Science and Technology, pp.209-216, 1984.

[HKF89] Hatakeyama A., Kawaguchi H., Fujiawa H., "Processing of Synonym and Variant for Free Term Search", IPSJ 39th Zenkoku-Taikai, p.1077, 1989.

[HGH+89] Hosono K., Gotoh T., Harada T., et al., "Analysing Characteristics of Kanji and Katakana Scripts of Technical Terms in Science and Technology Fields", 2nd International Conference on Japanese Information in Science, Technology and Commerce, pp.472-484, 1989.

[INT+86] Ishiwata T., Nishimura H., Tanaka H., et al.,"Nihongo Joho-Shori [Japanese Information Processing]", edited by N. Takahashi, Kindai Kagaku-sha, Tokyo, 1986.

[ISN92] Iwabuchi T., Suda A., Nakata Y., "Automatic Indexing Method of Japanese Language on Full-Text Documents", IPSJ 44th Zenkoku-Taikai, 4, pp.95-96, 1992.

[Kam89] Kamio T., "Automated Indexing for Making of a Newspaper Article Database", JICST Joho Kanri, 32(4), pp.283-293, 1989.

[KHK88] Kaneda K., Hanada J., Kato K., "Analysis of Retrieval Term and Evaluation on and Registration of New Keywords for the In-house Technical Document Management System", JICST 25th Annual Meetimg on Information Science and Technology, pp.139-154, 1988.

[Kim87] Kimoto H., "Automatic Keyword Extraction by a Natural Language Processing", 1st Jinkou-Chinou Gakkai, pp.389-392, 1987.

[KMK80] Kinukawa H., Matsuoka H., Kimura M., "Japanese Sentence Analysis for Automatic Indexing", COLING-80, pp.514-519, 1980.

[MT92] Masuoka T., Takubo Y., "Kiso Nihongo Bunpou [Basics of Japanese Grammar]", Kuroshio, Tokyo, 1992.

[MKM+91] Matsumoto Y., Kurohasi S., Myoki Y., et al., "Riyousha Teigi Kanou-na Nihongo Keitaiso Kaiseki System, JUMAN Shiyou Setsumeisho [User's Guide for the JUMAN System, A User-Extensible Morphological Analyzer for Japanese]", Nagao Laboratory, Kyoto University, 1991.

[MUH+91] Morohashi M., Umeda S., Hosono K., et al., "Statistical Analysis of Japanese Text by Kanji Usage Frequencies and its Applications", 1991 ASIS Workshop on Natural Language Information Processing, 1991.

[Nak87] Nakamura T., "Automatic Adjustment for Okurigana Variation in Japanese Segmentation", IPSJ 34thZenkoku-Taikai, pp.1297-1298, 1987.

[Neg89] Negishi M., "Problems in Construction and Service of Full Text Databases, Discussed with a Case at National Center for Science Information System", IPSJ Jouhou-gaku Kiso 14-1, pp.1-8, 1989.

[Neg92] Negishi M., "Technology Trend in Full-text Database", IPSJ Joho-shori, 33(4), pp.413-420, 1992.

[OKT92] Ogawa R., Kikuchi Y., Takahasi K., "Recent Developments in Full-Text Database Technologies", IPSJ Joho-shori, 33(4), pp.404-412, 1992.

[Pea89] Pearl J., "Probabilistic Reasoning in Intelligent Systems", Morgan Kaufmann, CA, 1989.

[Sal86] Salton G., "Another Look at Automatic Text-Retrieval Systems", Communications of the ACM, 29(7), pp.648-656, 1986.

[SM83] Salton G., McGill M., "Introduction to Modern Information Retrieval", McGraw-Hill, New York, 1983.

[San86] Sano H., "Use of English in Japanese Technical Information", 49th ASIS Annual Meeting, pp.300-305, 1986.

[Sas89] Sasaki T., "Similar Expression Retrieval based on Meaning of Kanji Character", IPSJ 39th Zenkoku-Taikai, pp.742-743, 1989.

[Sas88] Sasaki Y., "An Automatic Query Construction Experiment", Proc. of the 25th Annual Meeting on Information Science and Technology, pp.43-48, 1988.

[Som85] Someya K., "Toward Building Databases in Japanese - View of Keywords", JICST 22nd Annual Meeting on Information Science and Technology, pp.37-46, 1985.

[TAK+81] Takano F., Araki K., Kaneko A., et al., "Japanese Keywords Automatic Selection System(JAKAS) -Development of Implementation System", JICST 18th Annual Meetimg on Information Science and Technology, pp.35-44, 1981.

[Tan89] Tanaka H., "Shizen-gengo Kaiseki no Kiso [Foundation of Natural Language Analysis]", Sangyo-Tosho, Tokyo, 1989.

[TOT83] Tanaka K., Osada K., Tsuchiya S., "Analysis of Katakana Strings in Scientific/ Technological Document Abstracts", IPSJ 26th Zenkoku-Taikai, pp.1205-1206, 1983.

[Tom89] Tomatsuri Y., "Discussion on a Program for Selecting Keywords Automatically out of Japannese Text - in the Case of Geographical Articles", IPSJ 1989 Information Science Symposium, pp.135-144, 1989.

[Tsu91] Tsunoda T., "Sekai no Gengo to Nihongo [World Languages and Japanese]", Kuroshio, Tokyo, 1991.

[Tur91] Turtle H. R., "Inference Network for Document Retrieval", Doctoral Dissertation, University of Massachusetts, 1991.

[TC91] Turtle H. R., Croft W. B., "Evaluation of an Inference Network-based Retrieval Model", ACM Transactions on Information Systems, 9(3), pp.187-222, 1991.

[Van79] Van Rijsbergen C.J., "Information Retrieval", Butterworths, London, 1979.

[Yam92] Yamada K., private letter.