Information Retrieval on Turkish Texts

FAZLI CAN [1], SEYIT KOCBERBER, ERMAN BALCIK
CIHAN KAYNAK, H. CAGDAS OCALAN, ONUR M. VURSAVAS

Bilkent Information Retrieval Group
Computer Engineering Department
Bilkent University
Bilkent, Ankara 06800, Turkey

This is a preprint of an article published in "Can, F. , Kocberber, S., Balcik, E., Kaynak, C., Ocalan, H. C., Vursavas, O. M. 'Information retrieval on Turkish texts.' *Journal of the American Society for Information Science and Technology*. Vol. 59, No. 3 (February 2008), pp. 407-421."
http://www3.interscience.wiley.com/journal/117946195/grouphome/home.html

Abstract
We study information retrieval (IR) on Turkish texts using a large-scale test collection that contains 408,305 documents and 72 ad hoc queries. We examine the effects of several stemming options and query-document matching functions on retrieval performance. We show that a simple word truncation approach, a word truncation approach that uses language dependent corpus statistics, and an elaborate lemmatizer-based stemmer provide similar retrieval effectiveness in Turkish IR. We investigate the effects of a range of search conditions on the retrieval performance; these include the scalability issues, query and document length effects, and the use of stopword list in indexing.

Keywords: agglutinative languages, experimentation, matching function, performance, scalability, stemming, stopword list, test collection construction, Turkish.

INTRODUCTION

With the Internet technology and the increase in the size of online text information and the globalization, information retrival (IR) has gained more importance especially in commonly used languages. Turkey is the 22nd largest economy (Anderson & Cavanagh, 2006) and the Turkish language is among the most commonly used 20 languages in the world (Grimes, 1996). However, Turkish IR is a field that has not gained much interest. This is partly due to non existence of standart IR test collections in Turkish. In this study among other things we aim to provide such a collection. Furthermore, working with an agglutinative language such as Turkish instead of a member of the Indo-European family is a real and important issue since there are many things to be done in such languages within the context of IR research and develoment. The commercial Web search engines such as Turkish specific ones and Google provide access for Turkish text, but they keep their search techniques as trade secrets. On the other hand, many applications, from personal information management to national security, need effective methods in various languages. We provide the first thorough investigation of information retrieval with a large-scale Turkish test collection. - A preliminary version of this study can be seen in (Can, Kocberber, Balcik, Kaynak, Ocalan, & Vursavas, 2006). - We examine the effects of several stemming algorithms and query-matching functions, and various system parameters on retrieval effectiveness.

[1] Corresponding author: Computer Engineering Department, Bilkent University, Bilkent, Ankara 06800, Turkey; Web: http://www.cs.bilkent.edu.tr/~canf/, e-mail: canf@muohio.edu, voice: +90 (312) 290-2613, fax: +90 (312) 266-4047.

The first component of IR research on effectiveness is the test collection. In IR, standard test collections follow the Cyril Cleverdon's Cranfield tests tradition of laboratory experiments and involve three components: a set of documents, a set of user information requests or topics, and a set of relevance judgments made by human assessors for each topic (Sparck Jones, 1981). (The internal representation of these information needs is referred to as queries. However, in the paper we refer to the the written forms of user information needs as queries. This is to prevent confusion, since as will be seen later, in the written form of the user information requests we have three fields and one of them is called topic.) Test collections facilitate reproducibility of results and meaningful effectiveness comparison among different retrieval techniques. They play an important role in advancing the state of the art in IR as proven by the TREC experiments (Voorhees, 2005; Voorhees, Harman, 2005). Our document collection *Milliyet*, which has been developed for this study, is about 800 MB in size, contains 408,305 news articles and 72 ad hoc queries written and evaluated by 33 assessors.

In effectiveness studies, stemming is a major concern (Harman, 1991). We compare the effects of four different stemming options on (Turkish) IR effectiveness. They are no stemming, simple word truncation, the successor variety method (Hafner & Weiss, 1974) adapted to Turkish, and a lemmatizer-based stemmer for Turkish (Altintas & Can, 2002; Oflazer, 1994). We investigate the IR effectiveness of these stemming options in combination with eight query-document matching functions. We also examine the impact of the use of a stopword list on effectiveness.

Since the performance of a search engine may not scale to large collections (Blair, 2002) we examine the scalability issues of our approach by testing on increasing large portions of our collection.

To cover a wide range of IR application environments, we analyze the effects of query lengths on retrieval performance. Since an important possible difference in IR environments is the query lengths, i.e., number of words used in queries. For example, in the Web environment most information seekers use only a few words in their queries (Jansen & Spink, 2006). However, the size of the requests sent to commercial information system (e.g., West Publishing's WIN) is usually greater than 2 or 3 words (Thompson, Turtle, Yang, & Flood, 1994). It is also observed that the number of words in queries varies depending on the application area or increases due to collection size growth (Blair & Maron, 1985). As time passes, Web users tend to use more words in their queries (Semel, 2006).

In a similar way, we study how effectiveness varies with document length. In some environments we may have short documents, e.g., image captions; while in others we may have long documents, e.g., full text of scientific papers. Different types of documents may have different retrieval characteristics (Robertson, 1981, p.26; Savoy, 1999).

We hypothesize that within the context of Turkish IR the following items would improve the system performance in terms of higher retrieval effectiveness:

- the use of a stopword list in indexing (since they eliminate noise words from query and document representation);
- the use of language specific stemming algorithms would scale better (since more accurate stems would reflect better document content and this would be more noticeable in larger collections);
- longer queries (since they provide better description of user needs);
- longer documents (since the document contents may become more precise as we increase document sizes).

Our experiments are designed to test these hypotheses. Compared to previous studies on Turkish IR, our study includes a large-scale collection and a variety of retrieval scenarios.

Our contributions are summarized as follows; in this study we

- construct the first large-scale Turkish IR test collection. Due to its size, we can argue that our results are generalizable. The publicly available version of this collection would provide an important positive impact on Turkish IR research by offering a common denominator. Such collections open door to research and development of language specific retrieval techniques for improved performance by comparative evaluation based on measurement (Voorhees, 2005);
- investigate the effects of numerous system parameters (stemming options, query-document matching -ranking- functions, collection size, query lengths, document lengths) on Turkish IR and provide valuable observations and recommendations for research and development.

In Table 1 we provide the meanings of the frequently used acronyms. It is followed by a review of related works. After that we provide a quick overview of the Turkish language and the stemmers we use in this study. Then we describe the experimental environment in terms of various stopword lists, query-document matching functions used for ranking documents, document collection and queries. The experimental results given in the following section includes the effectiveness measure, stemming, matching function and scalability issues, stopword list, query length and document length effects. We conclude the paper with a summary of our findings and some future research pointers.

TABLE 1. Frequently used acronyms and their meanings.

F3…F7	Fixed Prefix stemmers (with prefix length equal to 3 … 7)	MF1 … MF8	Matching Functions 1 … 8, see (Table 2 for definitions)
LM5	Lemmatizer-based stemmer with average stem length = 5	NS	No Stemming
LM6	Lemmatizer-based stemmer with average stem length = 6.58	SV	Successor Variety stemmer
LV	LM5 stemmer, for words with no lemma uses SV for stemming	--	--

RELATED WORK

Information retrieval studies on languages other than English are less common. An incomplete list of such studies includes the works of Larkey, Ballesteros, and Connell (2002) on Arabic; Kettunen, Kunttu, & Jarvelin (2005) on Finnish; Savoy (1999) on French; Braschler and Ripplinger (2004) on German; Tordai and Rijke (2006) on Hungarian; Asian, Williams, and Tahaghoghi (2004) on Indonesian; Popovic and Willett (1992) on Slovene; Figuerola, Gomez, Rodriguez, and Berrocal (2006) on Spanish; and Ahlgren and Kekalainen (2007) on Swedish. TREC involves limited non-English experiments, for languages such as Arabic, Chinese and Spanish (TREC, 2007; Voorhees, 2005). On the other hand, the Cross Language Evaluation Forum (CLEF, 2007) activity (Braschler & Peters, 2004) whose document collection consists of more than 1.5 million documents in several European languages is an important research effort with several achievements. Savoy (2006), for example, reports the effectiveness of various general stemming approaches for French, Portuguese, German and Hungarian using the CLEF test collections. In the NTCIR

evaluation campaign (NTCIR, 2007), the Chinese, Japanese, and Korean languages are being studied.

Effect of stemming on IR effectiveness is an important concern (Frakes & Baeza-Yates, 1992). The results are a mixed bag. For example, Harman (1991), in her attempts with several stemming algorithms for English, was unable to succeed in improving the retrieval effectiveness. A similar observation is reported for Spanish (Figuerola, et al., 2006). However, for example for German, Braschler and Ripplinger (2004) show the positive impact of stemming on retrieval effectiveness. Similarly, later studies on English show positive impact of stemming on retrieval effectiveness (Hull, 1996; Krovetz, 1993). Stemming is also an important issue in Turkish IR studies.

The earliest published Turkish IR study is by Köksal (1981) and uses 570 documents on computer science with 12 queries. It evaluates the effectiveness of various indexing and document-query matching approaches using recall-precision graphs. After experimenting with various prefix sizes, Köksal uses the first 5 characters (5-prefix) of words for stemming.

Solak and Can (1994) use a collection of 533 news articles and 71 queries. The stemming algorithm of the Solak-Can study is based on looking for a given word in a dictionary, deleting a character from the end of the word, and then performing a structural analysis. The study shows 0% to 9% effectiveness improvement (in terms of precision at 10, i.e., P@10) with seven different query-document matching functions (corresponding to our matching functions MF1 to MF7, defined in a later section).

Ekmekçioğlu and Willett (2000) use a Turkish news collection of size 6,289 documents and 50 queries. They stem only query words (document words are used as they are), and compare the retrieval effectiveness using stemmed and unstemmed query words. In their study, a stemming-based query is effectively an extension of the unstemmed (original) query with various words corresponding to query word stems. They justify not stemming the documents words by stating that roots of Turkish words are usually not affected with suffixes. Note, however, that no stemming for documents, depending on the term weighting scheme, can affect the term weights in documents and queries (Salton & Buckley, 1988). They show that using the OKAPI text retrieval system their stemmed queries provide about 32% more number of relevant documents than that of unstemmed queries at the retrieval cut of values of (i.e., top) 10 and 20 documents. Their stemmer employs the same lemmatizer (Oflazer, 1994) that we use in the lemmatizer-based stemmer algorithm in this work.

Sever and Bitirim (2003) describe the implementation of a system based on 2,468 law documents and 15 queries. First, they demonstrate the superior performance of a new stemmer with respect to two earlier stemmers (one of them is the Solak-Can stemmer mentioned above). Then they show that their inflectional and derivational stemmer provides 25% retrieval precision improvement with respect to no stemming.

Pembe and Say (2004) study the Turkish IR problem by using knowledge of the morphological, lexico-semantical and syntactic levels of Turkish. They consider the effects of stemming with some query enrichment (expansion) techniques. In their experiments, they use 615 Turkish documents about different topics from the Web and 5 long natural language queries. They use seven different indexing and retrieval combinations and measure their performance effects.

On the Web there are several Turkish Web search engines and search directories (Can, 2006). Their quality and coverage vary. Bitirim, Tonta, and Sever (2002) investigate the performance of four Turkish Web search engines using 17 queries and measure their retrieval effectiveness, coverage, novelty, and recency.

STEMMING METHODS FOR TURKISH

Turkish Language

In this study by Turkish we mean the language mainly used in the republic of Turkey. The other dialects of Turkish, such as Azeri Turkish, are not our concern. Turkish belongs to the Altaic branch of the Ural-Altaic family of languages. Some concerns about this classification can be seen in Lewis (1988). The Turkish alphabet is based on Latin characters and has 29 letters consisting of 8 vowels and 21 consonants. The letters in alphabetical order are **a**, b, c, ç, d, **e** f, g, ğ, h, **ı**, **i**, j, k, l, m, n, **o**, **ö**, p, r, s, ş, t, **u**, **ü**, v, y, and z (vowels are shown in bold). In some words borrowed from Arabic and Persian, the vowels "a", "i," and "u" are made longer or softer by using the character ^ (circumflex accent) on top of them. In modern spelling, this approach is rarely used. In our collection they occur in a few number of documents. The '(quotation mark) is used to delimit the suffixes added to the proper names like in "Ali'nin evi İstanbul'da" which means "The house of Ali is in İstanbul."

Turkish is a free constituent order language, i.e., according to text flow and discourse context at certain phrase levels, its constituents can change order and still be intelligible (Lewis, 1988). For example, the sentence "İstanbul Ankara'dan daha güzel." ("İstanbul is more beautiful than Ankara.") and the sentence "Ankara'dan İstanbul daha güzel.", which is an inverted sentence ("devrik cümle" in Turkish), have the same meaning with a slight difference in emphasis (Lewis, 1988).

Turkish is an agglutinative language similar to Finnish and Hungarian. Such languages carry syntactic relations between words or concepts through discrete suffixes and have complex word structures. Turkish words are constructed using inflectional and derivation suffixes linked to a root. Consider the following examples for two roots of type, respectively, "noun" and "adjective."

- Ev (house), evim (my house), evler (houses), evlerde (in houses), evlerim (my houses), evlerimde (in my houses), evimdeyken (while in my house).
- Büyük (large), büyükçe (slightly large), büyüklük (largeness).

The following is a Turkish word obtained from the verb type root "oku" which means "to read."

- Okutamayacakmışçasına (oku + t + ama + yacak + mış + çasına) (as if not being able to make [*them*] read).

In these examples the meaning of the roots are enriched through affixation of derivational and inflectional suffixes (the morphemes of the last example are shown and separated by + signs). In Turkish verbs can be converted into nouns and other forms as well as nouns can be converted into verbs and other grammatical constructs through affixation (Lewis, 1988).

In Turkish, the number of possible word formations obtained by suffixing can be as high as 11,313 (Hakkani-Tür, 2000, p.31). Like other agglutinative languages, in Turkish it is possible to have words that would be translated into a complete sentence in non-agglutinative languages such as English. However, as we illustrate later, people usually do not use such words in their queries.

Like English, nouns in Turkish do not have a gender and the suffixes do not change depending on word type. However, there are some irregularities in adding suffixes to the words. Since these irregularities affect the stemming, we give a few examples in the following. (Please refer to (Lewis, 1988) for more detailed information on the Turkish language and grammar.) To obey the vowel harmony different suffixes are used to obtain the same meaning. For example, "ev" (means house) and "yer" (means ground) take the "de"

suffix and become “evde” (means in the house) and “yerde” (means on the ground), while “dağ” (means mountain) and “bahar” (means spring) take the “da” suffix and become “dağda” (means on the mountain) and “baharda” (means in the spring). In some words the last consonant change with some suffixes. For example, with the suffix “a”, “ağa**ç**” (means tree) becomes “ağa**c**a” (means towards the tree) while with the suffix “da” the root does not change and becomes “ağa**ç**ta” (means “on the tree”). However, in this example please notice the transformation of “da” to “ta” due to the letter “ç.” In some word and suffix combinations the letters in the word may drop. For example, with the suffix “um” the boldface letter **u** drops in “bur**u**n” (means nose) and becomes “burnum” (means my nose).

In Turkish only regular use of prefixation is to intensify the meaning of adjectives (and less commonly of adverbs). Such as “dolu” (full) and “dopdolu,” “tamam” (complete) and “tastamam” (Lewis, 1988, pp. 55-56). Such intensive adjectives are more suitable for story telling, but not for news articles. Prefixation in old fashioned words (such as “bîperva” which means “fearless”) or prefixation coming from western languages such as “antisosyal” (anti social) are infrequent in the language.

Stemming Methods

We use four stemming methods in obtaining the indexing terms. They are (1) no stemming, so called “austrich algorithm,” (2) first *n*, *n*-prefix, characters of each word, (3) the successor variety method adapted to Turkish, and (4) a lemmatizer-based stemmer.

No Stemming: The no stemming (NS) option uses all words as an indexing term. The retrieval performance of this approach provides a baseline for comparison.

Fixed Prefix Stemming: The fixed prefix approach is a pseudo stemming technique. In this method, we simply truncate the words and use the first *n* characters of each word as its stem; words with less than *n* characters are used with no truncation. We experiment with F3 to F7 ($3 \leq n \leq 7$).

We include the fixed prefix method due to the observation that Turkish word roots are not much affected with suffixes (Ekmekcioglu & Willet, 2000). It is true that words in any language have roots with different lengths. Nevertheless, Sever and Tonta (2006) also suggest the use 5-, 6-, or 7-prefix for rapid and feasible Turkish IR system implementation. Their suggestion is intuitive and based on their observation that truncated and actual Turkish words display similar frequency distributions; however, they do not provide any IR experiments. As we indicated in the above section, in Turkish the use of prefixes is uncommon. Therefore, in most cases the fixed prefix approach would truncate words with no prefixes. Note that the fixed prefix approach is similar to the *n*-gram approach, but in a much simpler form, since in the *n*-gram approach *n*-prefix is one of the *n*-grams that can be produced for a given word (McNamee & Mayfield, 2004). For example, for the word “İstanbul” the F4 stemmer generates the string “ista” as the word stem, the 4-grams of the same word are “ista,” “stan,” “tanb,” “anbu,” and “nbul.” For the word “bir” (one), which contains 3 characters, the F4 stemmer generates the word itself as its stem; similarly for this word we have only one string generated by the 4-gram approach and it is again the word itself.

Successor Variety Stemming: Successor Variety (SV) algorithm determines the root of a word according to the number of distinct succeeding letters for each prefix of the word to be stemmed can have in a large corpus (Frakes & Baeza-Yates, 1992; Hafer & Weiss, 1974). It is based on the intuition that the stem of a word would be the prefix at which the maximum successor variety is observed. For the working principles of the algorithm please refer to the example provided in (Frakes & Baeza-Yates, 1992, p. 135). Our SV implementation chooses

the longest prefix corresponding to the highest SV value (note that the same SV value can be observed for various prefix sizes), since longer stems would have a better reflection of the meaning of the complete word. Our SV algorithm directly returns the words that have length less than four characters without applying the SVprocess.

Our SV algorithm implementation has further adaptations to Turkish. In Turkish, when a suffix is used, a letter may change into another one, or may be discarded. For instance, the change of "**ç**" to "c" in our above example of "ağa**ç**" and "ağa**c**a" is an example of letter transformation. The above example of "bur**u**n" and "burnum" illustrates the second case since the letter "u" drops. Another feature that can affect stemmers for Turkish is that there are compound words obtained by concatenating two words. For example, "hanımeli", which is a flower name, contains the words "hanım" and "eli", which respectively mean "lady" and "hand." Finding "hanım" as the stem of "hanımeli" would be meaningless.

Our current SV algorithm only handles the letter-to-letter transformations, which are the far most frequently seen characteristic among the above ones. When the algorithm detects a possibility of transformation, it checks for the probability if that transformation exists. The probabilities are calculated by using the distribution of corpus words that are related to transformations. If it is greater than a threshold value, then the prefix under consideration contributes to the SV count of the corresponding non-transformed prefix. For example, for "ağac**a**", letter 'a' marked in bold, contributes to the successor variety value of the stem (prefix) "ağaç".

Lemmatizer-based Stemming: A lemmatizer is a morphological analyzer that examines inflected word forms and returns their base or dictionary forms. It also provides the type (part of speech, POS, information) of these matches, and the number and type of suffixes (morphemes) that follow the matches. We use the morphological analyzer presented in (Oflazer, 1994). Note that lemmatizers are not stemmers, since the latter obtains the root in which a word is based; in contrast, a lemmatizer tries to find the dictionary entry of a word. Being an agglutinative language, Turkish has different features from English. For English, stemming may possibly yield "stems" which are not real words. Lematization on the other hand tries to identify the "actual stem" or "lemma" of the word, which is the base form of the word that would be found in the dictionary. Due to the nature of English, sometimes words are mapped to lemmas, which apparently do not have any surface connection as in the case of *better* and *best* being mapped to *good*. However, Turkish does not have such irregularities and it is always possible to find the "stem" or "lemma" of any given word through application of grammar rules in removing the suffixes. For this reason, throughout the paper, we prefer the word "stemming" over lemmatization; as it is more commonly used, and the algorithm we use internally identify the suffixes and remove them in the stemming process.

In the lemmatization process in most of the cases we obtain more than one result for a word, in such cases the selection of the correct word stem is done by using the following steps (Altintas & Can, 2002): (1) Select the candidate whose length is closest to the average stem length for distinct words for Turkish; (2) If there is more than one candidate, then select the stem whose word type (POS) is the most frequent among the candidates.

For the above algorithm, we need to know the average type stem length, which is experimentally found as 6.58 by Altintas & Can (2002) by using a disambiguated large corpus, and the word type, i.e., POS, frequencies in Turkish. They showed that the success rate of the algorithm in finding the correct stems is approximately 90%. Having a result of around 90% may be imperfect, but acceptable.

In this study, as the first algorithm parameter, i.e., the average stem length, we use 6.58 and 5. We use the length 5 since as will be illustrated in the experimental results section, 5-

prefix provides the best effectiveness among the *n*-prefix methods. These two versions of the algorithm are referred to as LM5 and LM6. For various items, including misspelled and foreign words, which cannot be analyzed by the lemmatizer, in an additional **L**M5 version, we use the S**V** method for such words, this crossbreed is referred to as LV.

STOPWORD LIST

In IR, a stopword list contains frequent words that are ineffective in distinguishing documents from each other. In indexing, it increases the storage efficiency by eliminating the posting lists of such words from the indexing process. However, with the decreasing cost of the secondary storage this issue has lost its importance (Witten, Moffat, & Bell, 1999). Dropping stopwords can also increase the query processing efficiency. The construction of a stopword list involves various, sometimes arbitrary, decisions (Savoy, 1999).

In the IR literature it is possible to find stopword lists with different lengths even for a given specific language. For English, Fox (1990) suggests a stopword list of 421 items, and the SMART system uses a list with 571 English "forms" (SMART, 2007), commercial information systems tend to adopt a very conservative approach with only a few stopwords. For example, the DIALOG system is using only 9 items (namely "an," "and," "by," "for," "from," "of," "the," "to," and "with") (Harter, 1986).

In this study we use three stopword lists. We first used a semi-automatic stopword generation approach. For this purpose, we ranked all words according to their frequencies (i.e., total number of occurrences in all documents). Then, we determined a threshold value so that the words whose frequencies above the threshold became a stopword candidate. In the manual stage, we removed some words selected so far, since they have information value (e.g. "Türkiye," and "Erdoğan" –current prime minister–). We also added some variations of the selected words to the list and all letters of the Turkish alphabet and the letters q, w, and x. The semi-automatically generated stopword list contains *147* words and is given in Appendix, Table A.1, in alphabetical order to see variations of some words. The stopword list cover 14% of all word occurrences in documents when no stemming is used.

We also experimented with a second stopword list and just used the top most frequent 288 words with no elimination. When we list the words, we observe that the top words cover a significant fraction of all word occurrences, but this coverage begins to disappear as we include more words to the stoplist. After observing this, we stop including such words to the list. This process gives 288 words and they cover 27% of all word occurrences. The second set is used to understand the retrieval effectiveness consequences of automatic construction of stopword lists in Turkish.

As an extreme case we also experimented with a short stopword list that contains the most frequent first ten words ("ve" "bir," "bu,", "da," "de," "için," "ile," "olarak," "çok,", and "daha," their meanings in order are "and," "a/an/one," "this," "too," "too," "for," "with," "time," "very," and "more"). The word "olarak" is usually used in phrases like "ilk olarak/for the first time," and "son olarak/ for the last time). These words cover 8% of all word occurrences.

MATCHING (RANKING) FUNCTIONS

Assigning weights to terms in both documents and queries is an important efficiency and effectiveness concern in the implementation of IR systems (Cambazoglu & Aykanat, 2006; Lee, Chuang, & Seamons, 1997). In this study for term weighting we use the *tf.idf* model.

Term weighting has three components: term frequency component (TFC), collection frequency component (CFC), and normalization component (NC). The weights of the terms of a document and a query (denoted by w_{dk} and w_{qk}, $1 \leq k \leq$ *no. of terms*, i.e., the number of terms used in the description of all documents) are obtained by multiplying the respective weights of these three weighting components. After obtaining the term weights, the matching function for a query (Q) and a document (Doc) is defined with the following vector product (Salton & Buckley, 1988).

$$similarity(Q, Doc) = \sum_{k} w_{dk} \cdot w_{qk}$$

In ranking-based text retrieval, documents are ranked according to their similarity to queries. The weight w_{dk} of term t_k in Doc is defined by the following product (TFC x CFC x NC). The three possibilities for TFC are symbolized by *b*, *t*, *n* and corresponds to *tf* of well-known *tf.idf* indexing approach in IR (Witten, et al., 1999). In TFC, *b* is binary weight; in this case ignore the term frequency and take TFC= 1; *t* is term frequency, which means TFC is equal to the number of occurrences of t_j in d_i; *n* is the augmented normalized term frequency and defined as (0.5 + 0.5 x t / maxt) where maxt is the maximum number of times any term appears in d_i.

The three possibilities for CFC are denoted by *x*, *f*, *p*. In CFC, *x* indicates no change, i.e., take CFC= 1. The symbol *f* indicates the inverse document frequency, *idf*, and in this study it is taken as $\ln(N/t_{gj}) + 1$ for document and query terms, N is total number of documents in the collection, and t_{gj} is the number of documents containing t_j. The symbol *p* is the probabilistic inverse collection frequency factor and it is similar to *f* both in terms of definition and performance (Salton & Buckley, 1988). We did not use it in our experiments.

For normalization, i.e., the NC component, there are two possibilities denoted by *x* and *c*. The symbol *x* means no change, i.e., take NC= 1; *c* means cosine normalization where each term weight (TFC x CFC) is divided by a factor representing Euclidian vector length. The normalization of query terms is insignificant since it does not change the relative ranking of documents.

TABLE 2. Matching (ranking) functions used in the experiments.

Matching Function	**MF1**	**MF2**	**MF3**	**MF4**	**MF5**	**MF6**	**MF7**
Meaning	txc.txx	tfc.nfx	tfc.tfx	tfc.bfx	nfc.nfx	nfc.tfx	nfc.bfx

Various combinations of the term weighting components yield different matching functions, i.e., index structures as given in Table 2. For example, MF1, i.e., the combination txc (TFC= *t*, CFC= *x*, and NC= *c*) and *txx*, respectively, for documents and queries yields the well known cosine function with no *idf* component, it simply uses document and query vectors as they are. The combinations *tfc*, *nfc*, for documents and *nfx*, *tfx*, *bfx* for queries have been determined to result in better IR performance. These provide us with the six different matching functions (MF2-MF7): "*tfc.nfx*," "*tfc.tfx*," "*tfc.bfx*," "*nfc.nfx*," "*tfc.tfx*," and "*nfc.bfx*." These are similar but different document-query matching functions and highly recommended by Salton and Buckley (1988). Table 2 shows these seven combinations, i.e., the query matching functions used in our experiments. These are the matching functions that we have used in our previous studies for information retrieval in English (Can & Ozkarahan, 1990) and Turkish (Solak & Can, 1993) texts.

Additionally, we used MF8 (Long & Suel, 2003; Witten et al., 1999). MF8 calculates matching value for document d_j for the search query Q as follows:

$$MF8 = \sum_{t \in Q} \left((1 + \ln f_{dt}) / \sqrt{D} \right) \cdot \left(f_{qt} \cdot \ln(1 + (N / f_t)) \right)$$

where f_{dt} is the frequency of term t in document d_j, D is the total number of term occurrences in d_j, f_{qt} is the frequency of term t in the query Q, as defined previously N is the total number of documents, and f_t is the frequency of term t in the entire document collection. Note that MF8 is especially suitable for dynamic environments, since in dynamic collections one can easily reflect the effects of *idf* to the term weighting scheme via query term weights (the second item of the MF8 formula).

TEST COLLECTION

Our document collection contains 408,305 documents; they are the news articles and columns of five years, 2001 to 2005, from the Turkish newspaper *Milliyet* (www.milliyet.com.tr). The size of the *Milliyet-2001-2005* (for brevity *Milliyet*) collection is about 800 MB and without stopword elimination each document contains 234 words (tokens) on the average. It contains about 95.5 (89.46 alphabetic, 4.66 numeric, and 1.36 alphanumeric) million words before stopword elimination. We converted all uppercase letters to their smallcase equivalents and used UTF-8 for character encoding. The ‘(quotation mark) and – (dash) are considered as part of a word unless they are the last character. The letters “a”, “i,” and “u” with the character ^ on top of them are taken as distinct letters than their counterparts. The average word (token) length is 6.90 characters.

TABLE 3. Indexing information with the stopword list provided in the Appendix Table A.1

Information	**NS**	**F5**	**F6**	**SV**	**LV**
Total no. of terms	1,437,581	280,272	519,704	418,194	434,335
Avg. no. of terms/doc.	148	124	132	119	117
Avg. term length	9.88	4.82	5.66	7.23	7.24
Median term length	9	5	6	7	7
Min. term length	2	2	2	2	2
Max. term length	50*	5	6	46*	46*
Std. dev. for term length	3.58	0.50	0.69	2.74	2.71
No. of posting elements (millions)	60	51	54	48	48
Storage efficiency wrt NS	N/A	15%	10%	20%	20%

* Do not correspond to actual words, due to errors such as missing blank spaces, etc.

The indexing information with different stemmers, using the Appendix Table A.1 stopword list, is shown in Table 3. After stopword elimination on the average in each document contains 201 words (tokens). When NS is used on the average each document contains 148 terms (unique words, i.e., types). In this table total number of terms indicates the number of unique word in the collection. The table also contains various information on term lengths. The longest meaningful word in the whole collection is the word “Danimarkalılaştıramadıklarımızdan.” It contains 33 characters and means, “he (she) is one of those who we were unable to convert to Danish.” However, note that such words are uncommon as illustarted by Figure 1. This figure shows the total number of unique terms with a certain length for all stemming options. For F5 and F6 there is no observation after 5 and 6 characters, respectively.

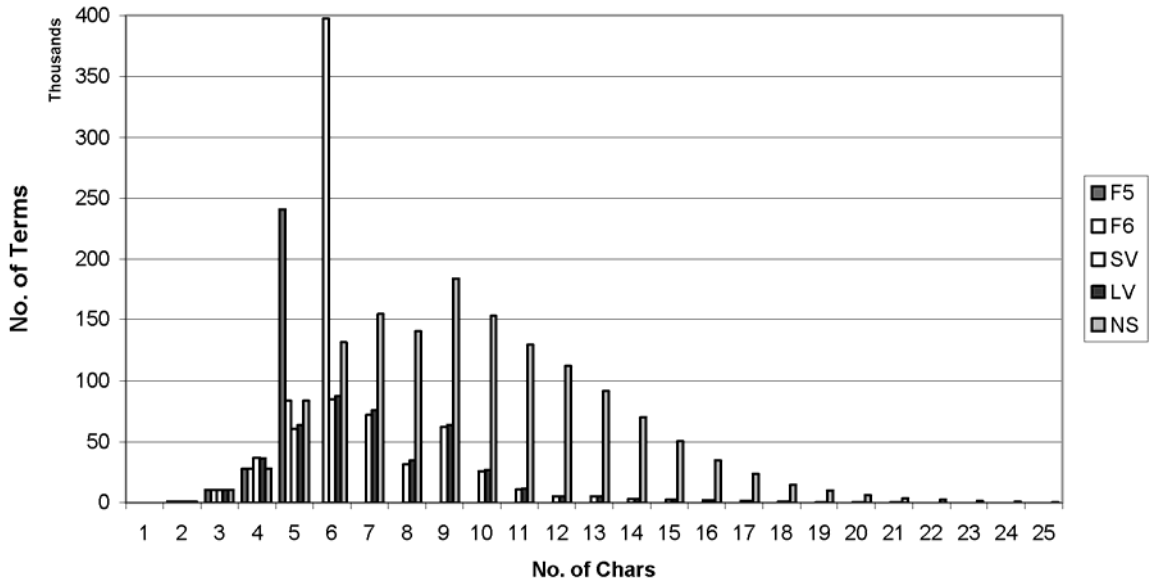

FIG. 1. Term frequency vs. term length (in characters) for all stemmers.

The posting lists sizes with different stemming options, in terms of <word, word frequency> pair, are also shown in Table 3. These values for NS, F5, and LV are, respectively, 60, 51, 48 million entries. This means that F5 and LV provide 15% (51 vs. 60) and 20% (48 vs. 60) storage efficiency with respect to NS. Without stopword elimination the posting lists contain 67 million entries.

The queries are written and evaluated according to the TREC approach by 33 native speaker assessors. The original query owners do the evaluation using binary judgment. Relevant documents are identified by examining the union of the top 100 documents of the 24 possible retrieval combinations, "runs," of the 8 matching functions and the stemmers NS, F6, and SV that we had at the beginning of our experiments.

For determining the relevant documents of the queries the pooling concept is used (Zobel, 1998). During evaluation, the query pool contents are presented to the assessors (i.e., the query owners) in random order and the rest of the collection is assumed to be irrelevant. The assessors use a Web interface for query evaluation. All assessors are experienced Web users: graduate and undergraduate students, faculty members, and staff. They are allowed querying any information need that they choose, and are not required to have an expertise on the topic that they pick. The query subject categories and the number of queries in each category are shown in the Appendix, Table A.2.

A typical query is a set of words that describe a user information need with three fields: topic (a few words), description (one or two sentences), and narrative (more explanation). The topics of all queries and the number of relevant documents for each query are listed in the Appendix, Table A.3. The query topics cover a wide range of subjects and distributed to all five years covered by the collection.

During pooling, for the construction of the query vectors, only the topic and description fields have been used. The average pool size and relevant documents per query, are 466.5 and 104.3 respectively. We have 72 queries after eliminating 20 queries with too few ($\leq$ 5% of its pool) or too many ($\geq$ 90% of its pool) relevant documents in their pools: it is known that such queries are not good at discriminating retrieval systems (Carterette, Allan, & Sitaraman, 2006). A typical query evaluation takes about 130 minutes. The total number of documents and unique documents identified as relevant are 7,510 and 6,923, respectively.

In the rest of this paper, we will refer the query forms made out of *Topic* as Q_S (short query), *Topic+Description* as Q_M (medium length query), and *Topic+Description+Narrative* as Q_L (long query). Tables 4 and 5 respectively show the query and query word length statistics for the queries. Note that from short to long queries the variety of the words (number unique words) and the average length of both words and unique words increase.

The most frequently used top 10 words in the (Q_M) queries are "türkiye'de," "etkileri," "üzerindeki," "türk," "gelen," "son," "türkiye," "avrupa," "meydana," "şiddet." These words account for 10.26% of all word occurrences out of 1004 query words. The frequently used query words are short; actually, this is true for all query words. They are not like extreme Turkish word examples; such as "Avrupalılaştırılamayabilenlerdenmişsiniz," which means, "you seem to be one of those who may be incapable of being Europeanized" (Ekmekcioglu & Willet, 2000). From Q_S to Q_L query words become slightly longer (see Table 5) as users are given the opportunity of expressing their information needs in more detail in narrative form.

TABLE 4. Query statistics.

Entity	Min.	Max.	Median	Avg
Pool Size (no. of unique documents)	186	786	458	466.5
No. of relevant documents in query pools	18	263	93	104.3
Query evaluation time* (minutes)	60	290	120	132.4
No. of unique words in Q_S**	1	7	3	2.89
No. of unique words in Q_M**	5	24	11	12.00
No. of unique words in Q_L**	6	59	26	26.11

* Entered by participants, ** With stopwords.

TABLE 5. Query word statistics.

Entity	Q_S	Q_M	Q_L
No. of Words	208	1004	2498
No. of Unique Words	182	657	1359
Avg word Length	7.03	7.57	7.62
Avg Unique Word Length	7.00	7.75	8.04

EXPERIMENTAL RESULTS

Effectiveness Measures

Precision and recall are the most commonly known effectiveness measures in IR. They are, respectively, defined as proportion of retrieved documents that are relevant, and proportion of relevant documents retrieved. Recall is difficult to measure in real environments. Precision at top 10 and 20 documents (P@10, P@20) are sometimes the preferred measure because of their simplicity and intuitiveness. Furthermore, in the Web environment search engine users usually only look at the top two pages and P@10 and P@20 reflect the user satisfaction. However, mean average precision (MAP), which is the mean of the average precision value when a relevant document is retrieved is considered as a more reliable measure for effectiveness (Buckley & Voorhees, 2004; Sanderson & Zobel, 2005; Zobel, 1998).

In our case, as we indicated before the query pools (used in determining the relevant documents) are constructed by using the stemmers NS, F6, and SV. However, these relevance judgements are also used for the evaluation of systems, i.e., "stemmer and matching function" combinations that are not used in the construction of the query pools. This may disadvantage such systems due to a possible bias. For such cases Buckley and Voorhees (2004) have introduced a new measure called "binary preference" or *bpref* that ignores the documents not evaluated by users. For this reason, for performance measurement we use the *bpref* measure. We use the trec-eval package version 8.1 for obtaining the effectiveness measures. When necessary we conduct two-tailed t-tests for statistical analysis using an alpha level of 0.05 for significance.

Selection of Stemmers for Overall Evaluation

In order to streamline the overall evaluation process, first we determine the best representative of the fixed prefix and the lemmatizer-based methods. Table 6 shows the assessment of all fixed prefix methods according to different effectiveness measures with the matching function MF8 that gives the best performance for all stemmer and matching function combinations. In terms of *bpref*, F4 and F5 are better than the rest (e.g., 5% to 6% better than F3). For choosing only one of these matching functions we also consider the MAP, P@10, and P@20 values. In terms of MAP measure, the performance of F5, which is not used in the construction of query pools, is 5% better than that of F6 (the only fixed prefix method used in constructing the query pools). The same is approximately true for F4. According to the MAP results, F3 and F7 are obvious losers. The *bpref* and MAP values of F4 and F5 are close to each other; on the other hand, P@10 and P@20 values of F5 are about 5% higher than that of F4. Due to these observations, we use F5 as the representative of the fixed prefix stemmers. However, note that the two-tailed t-test results indicate no statistically significant difference between F4 and F5 with MF8 using P@10 and P@20 individual query results.

In a similar fashion, LM5 is slightly better than LM6. As a lemmatizer-based stemmer, we also have LV that takes advantage of LM5 and SV. LV shows slightly, but not significantly, better performance than LM5. As a result, for the final analysis we have NS, SV, F5 (the representative for the fixed prefix methods), and LV (the representative for the lemmatizer methods).

TABLE 6. Various effectiveness measure results with MF8 and Q_M for fixed prefix methods F3-F7.

Method	*bpref*	MAP	P@10	P@20
F3	.4120	.3134	.5139	.4757
F4	.4382	.4013	.5625	.5361
F5	.4322	.4092	.5917	.5653
F6	.4014	.3885	.5667	.5382
F7	.3901	.3658	.5556	.5181

Overall Evaluation: Effects of Stemming and Matching Functions

Table 7 shows the performance of NS, SV, F5, and LV (we also include LM5 for comparison with LV) in terms of *bpref*. The table also shows the percentage performance improvement of LV, SV, and F5 with respect to NS; and the improvement provided by LV with respect to SV and F5. For easy comparison *bpref* values of NS, F5, SV, and LV are shown as bar charts in Figure 2. In terms of MF8, which provides the best performance with all matching functions, the stemming methods F5, LV, and SV, respectively, provide 32.78%, 38.37%, and 32.23% better performance than that of NS. These are all statistically significant improvements ($p < 0.001$).

In our IR experiments, the most effective stemming method is LV. The performance comparison of F5, LV, and SV in terms of MF8 (Table 7) shows that LV is slightly, 4.65% and 4.21%, better than SV and F5 (please look at the LV/SV and LV/F5 columns); however, these differences are statistically insignificant. The SV method and the simple prefix method F5 are also effective, but not as good as LV; and F5 is slightly better than SV. The LV stemmer performs slightly better than F5; however, the difference is statistically insignificant.

TABLE 7. *Bpref* values of NS to LM5 and % improvement of LV wrt. NS (LV/NS) to LV wrt. F5 (LV/F5) using query form Q_M and matching functions MF1 through MF8.

	bpref					Percentage of Improvement				
MF	**NS**	**F5**	**SV**	**LV**	**LM5**	**LV/NS**	**SV/NS**	**F5/NS**	**LV/SV**	**LV/F5**
MF1	.2452	.3108	.3046	.3339	.3275	36.18%	24.23%	26.75%	9.62%	7.43%
MF2	.3124	.3961	.4096	.4175	.4095	33.64%	31.11%	26.79%	1.93%	5.40%
MF3	.3045	.3823	.3908	.4054	.3992	33.14%	28.34%	25.55%	3.74%	6.04%
MF4	.3099	.3905	.4030	.4122	.4045	33.01%	30.04%	26.01%	2.28%	5.56%
MF5	.2849	.3764	.3663	.3890	.3805	36.54%	28.57%	32.12%	6.20%	3.35%
MF6	.2982	.3883	.3678	.3908	.3847	31.05%	23.34%	30.22%	6.25%	0.64%
MF7	.2692	.3532	.3477	.3734	.3642	38.71%	29.16%	31.20%	7.39%	5.72%
MF8	.3255	.4322	.4304	.4504	.4447	38.37%	32.23%	32.78%	4.65%	4.21%
Col.Avg.	.2854	.3715	.3675	.3922	.3861	35.08%	28.38%	28.93%	5.26%	4.79%

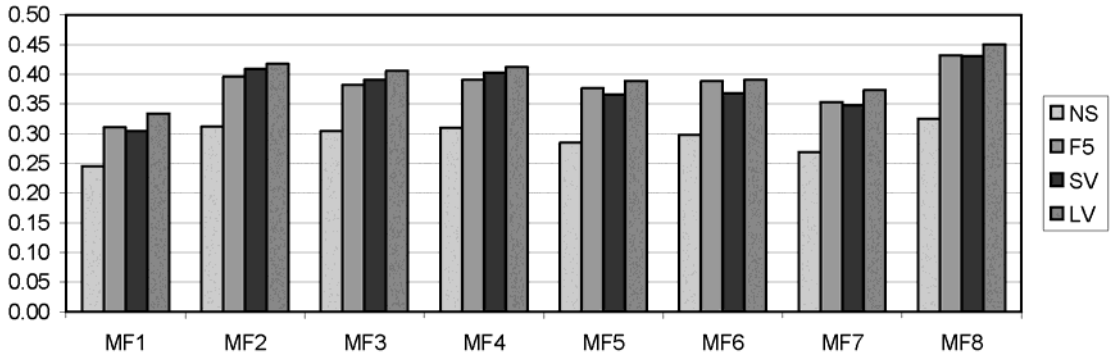

FIG. 2. *Bpref* values of NS, F5, SV, and LV with matching functions MF1-MF8 using Q_M.

The comparisons of F5, LV, and SV with the average results of the matching functions MF1 through MF8 (using the corresponding eight average *bpref* values given in the columns of Table 7) show no statistically significant difference between F5 and SV, but significant difference between F5 and LV, and SV and LV using two-tailed t-tests ($p < 0.001$). Please note that these comparisons are based on the average *bpref* values listed in Table 7, i.e., the column-wise comparison of the eights *bpref* values (corresponding to MF1 through MF8) for F5, LV, and SV. However, comparison of these methods by using the individual *bpref* values of the queries for a given matching function (for example, MF8 results with F5 and LV) show no statistically significant difference using two-tailed t-tests. These results show that a simple word truncation approach (F5) and a careful word truncation approach that uses language dependent corpus statistics (SV) and a sophisticated lemmatizer-based stemmer (LV) provide comparable retrieval effectiveness. This is conceptually in parallel with the findings of a study on another agglutinative language: Kettunen, Kunttu, & Jarvelin (2005) show that for Finnish, a lemmatizer and a simple (Porter like) stemmer provide retrieval environments with similar effectiveness performances.

Our results show that MF1 is the poorest performer. This can be explained by the lack of the *idf* component in its definition. MF2 outperforms the other matching functions, except MF8. Similar results have been reported elsewhere (Can & Ozkarahan, 1990) about the performance of MF1-MF7. The relative performances of these matching functions are consistent in this and the aforementioned study. Our results show that MF8, which involves no (document) term re-weighting due to addition or deletion of documents, gives the best performance. This has practical value in dynamically changing real environments.

In the following sections, for performance comparison we use the results of MF8 (the matching function that provides the best performance), and only consider NS, F5, and LV, due to comparable performances of F5 and SV.

Effects of Stopword List on Retrieval Effectiveness

In this section, we analyze the effects of stopword list on retrieval effectiveness. In the first set of experiments we measure *bpref* values using the semi manually constructed stopword list (of Appendix Table A.1) and without using a stopword list. The results presented in Table 8 along with two-tailed t-tests show that stopword list have no significant impact on performance. Note that the assessors (query owners) are told nothing about the use of frequent Turkish words; nevertheless, such words have not been used heavily in the queries. For example, in Q_M on the average a query contains 1.74 stopwords.

TABLE 8. *Bpref* values using Q_M with (NS, F5, LV) and without (NS', F5', LV') a stopword list.

NS	NS'	F5	F5'	LV	LV'
0.3255	0.3287	0.4322	0.4330	0.4504	0.4524

In the above approach, we use the stopword list to eliminate words before entering them to the stemmers. As an additional experiment, we have used the stopword list after stemming. For this purpose, we first used F5 stemmer to find the corresponding stem and after that, we search the stemmed word in the stemmed stopword list. The experiments again show no statistically significant performance change.

In order to observe the possible effects of automatic stopword list generation we also use the automatically generated stopword lists of the most frequent 288 words, and 10 words. The IR effectiveness performance with them is not statistically significantly different from the case with no stopword list.

From these observations, we conclude that the use of a stopword list have no significant effect on Turkish IR performance. However, note that this may be a result of the *tf.idf* model we use. For example, Savoy (1999) reports experiments on French text in which stopword list does not have any influence with the *tf.idf* model; but has influence with the OKAPI model. Our results are consistent with his observations.

Scalability

Scalability is an important issue in IR systems due to the dynamic nature of document collections. For this purpose, we have created eight test collections in 50,000 document increments of the original test collection. The first one contains the initial 50,000 documents (in temporal order) of the *Milliyet* collection, the second one contains the first 100,000 documents and is a superset of the first increment. The final step corresponds to the full version of the document collection.

For evaluation, we use the queries with at least one relevant document in the corresponding incremental collection. For example, for the first 50,000 documents, we have 57 active queries, i.e., queries with at least one relevant document in the first 50,000 documents. Table 9 shows that each increment has similar proportional query set characteristics; for example, median number of relevant documents per query increases approximately 10 by 10 (11.0, 21.5, 34.0, etc.) at each collection size increment step. This means that experiments are performed in similar test environments.

TABLE 9. Query relevant document characteristics for increasing collection size.

No. of Docs	No. of Active Queries	Total No. of Unique Relevant Docs.	Avg. No. of Rel. Doc/Query	Median No. of Rel. Doc/Query
50,000	57	719	10.72	11.0
100,000	62	1380	21.08	21.5
150,000	63	2014	30.55	34.0
200,000	64	2944	44.33	45.5
250,000	68	3764	56.51	56.5
300,000	70	4794	71.45	66.0
350,000	71	5725	86.29	79.0
408,305	72	6923	104.30	93.0

Understanding the retrieval environments in more detail might be of interest. The characteristics of the collections as we scale up are shown graphically in Figures 3 and 4. Figure 3 shows that the number of unique words increases with the increasing collection size; however, F5 and LV show saturation in the increase of unique words as we increase the number of documents, and this is more noticeable with F5. Figure 4 shows that the number of postings (i.e., <document number, term weight> pairs or tuples) in the inverted files of NS, F5, and LV linearly increases as we increase the collection size. The graphical representation of posting list sizes of this figure indicates that with NS we have many short posting lists.

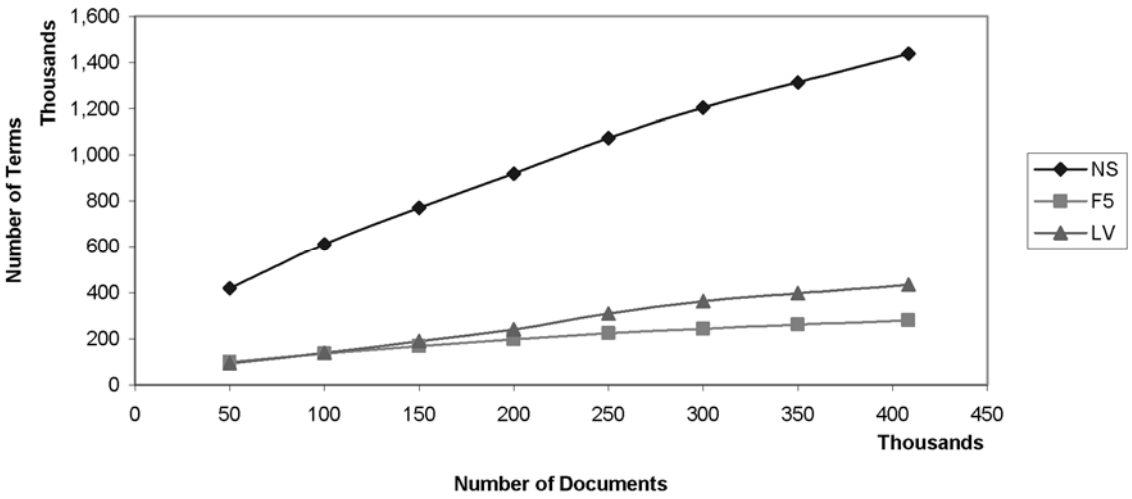

FIG. 3. Indexing vocabulary size vs. collection size.

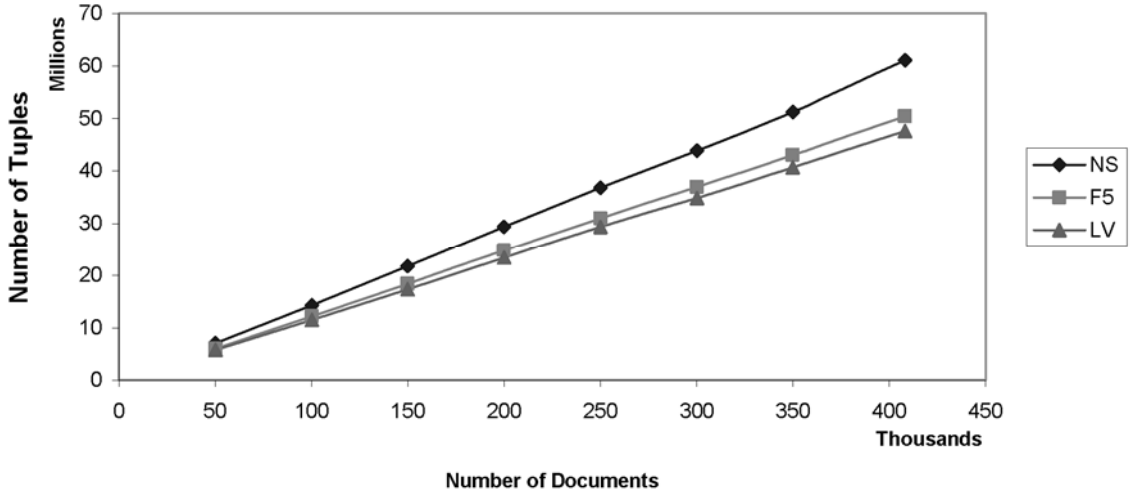

FIG. 4. Number of posting list tuples vs. collection size.

The performance of NS, F5, and LV in terms of *bpref* as we scale up the collection is presented in Figure 5. With the first increment, we have a relatively better performance with respect the performances of the next three steps. In the second incremental step, i.e., with 100,000 documents, we have a decrease in performance, then we have a tendency of performance increase and beginning with 250,000 documents, we have a steady retrieval performance. This can be attributed to the fact that after a certain growth document collection characteristics reaches to a steady state.

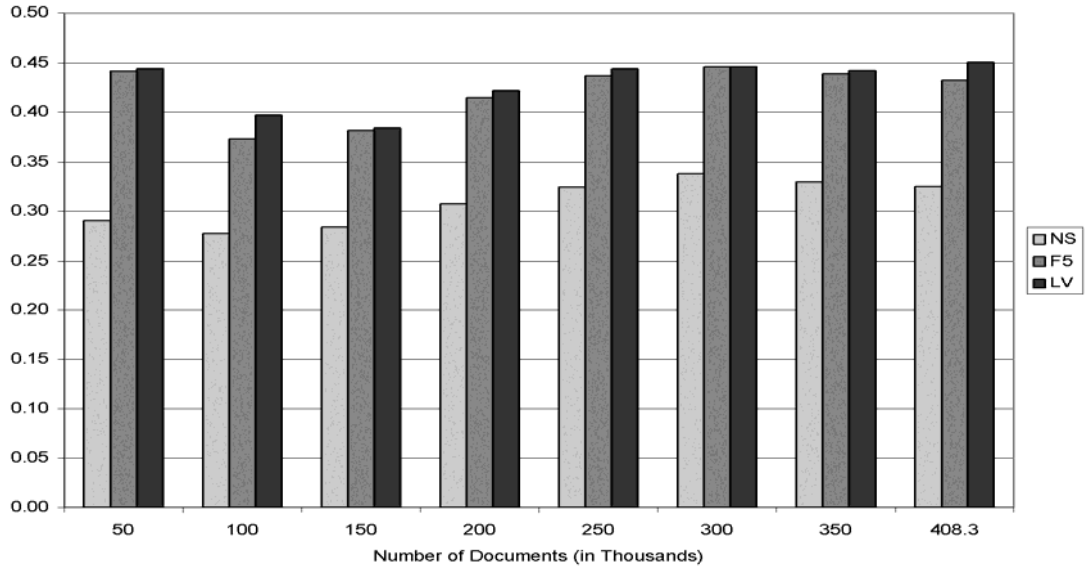

FIG. 5. *Bpref* values with MF8 for NS, F5, and LV using Q_M as collection size scales up.

In this work, our concern is the relative performances of NS, F5 and LV. We see that matching functions show steady relative effectiveness performances. Contrary to our initial hypothesis, the LV stemmer, which is designed according to the language characteristics, shows no improved performance as the collection size increases: simple term truncation method of F5 and LV are compatible with each other in terms of their retrieval effectiveness performances throughout all collection sizes. LV provides slightly better, but statistically insignificant performance improvement with respect to F5. However, the performance of F5 and LV with respect to NS is statistically significantly different (in all cases $p < 0.001$).

Query Length Effects

In an IR environment, depending on the needs of the users, we may have queries with different lengths. For this reason, we analyze the effects of query lengths on effectiveness. The query types according to their lengths are described in our test collection discussion (please refer to Tables 4 and 5).

The experimental results are summarized in Figure 6. The figure shows that as we go from Q_S to Q_M we have a statistically significant ($p < 0.01$) increase in performance using F5 and LV. Improvements in effectiveness for them, respectively, are 14.4%, and 13.5%. The tendency of performance increase can be observed as we go from Q_M to Q_L, but this time the increase is statistically insignificant. For all query cases, under the same query form the performance difference of F5 and LV is statistically insignificant. However, the performance difference of these stemmers with respect to NS is statistically significant ($p < 0.001$). (The findings stated in the last two sentences were presented before for only Q_M, here we provide the observations for the other two query forms Q_S and Q_L.) In terms of NS, the performance increase is first 6.23% and then 14.59% as we increase the query lengths incrementally. The second increase is statistically significant ($p < 0.01$). In other words, NS gets more benefit from query length increase. Also, the negative impact of not being stemmed is partly recovered with the increase in query length. From the experiments, we observe that there is no linear relationship between query lengths and retrieval effectiveness. That is, as we increase the query length, first we have an improvement and after a certain length increase this effectiveness increase tends to saturate. However, the NS approach improves its performance as we increase the query length.

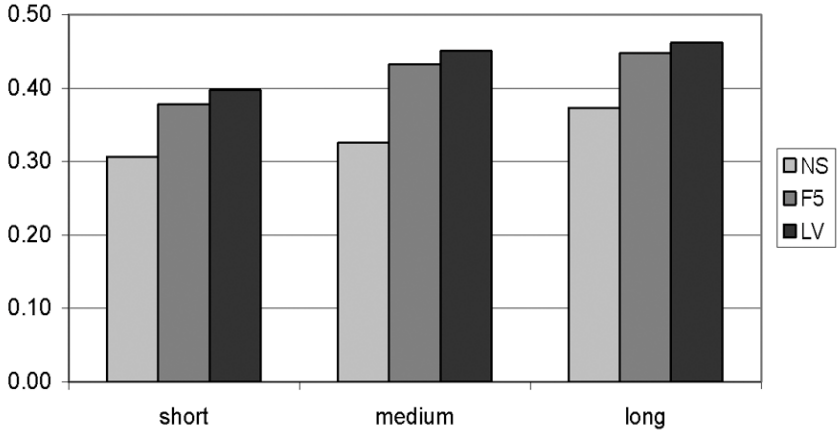

FIG. 6. *Bpref* values with various query lengths using MF8.

The effectiveness improvement can be attributed to the fact that longer queries are more precise and provide better description of user needs. Similar results are reported in other studies regarding the effects of increasing query lengths. For example, Can, Altingovde, & Demir (2004) report similar results for increasing query lengths with the Financial Times TREC collection.

Document Length Effects

In different application environments, it is possible to have documents with different lengths. For this reason we divided the *Milliyet* collection according to document lengths and obtained three sub-collections that consist of short (documents with maximum 100 words), medium length (documents with 101 to 300 words), and long documents (documents with more than 300 words). In a similar fashion, we divided the relevant documents of the queries among these sub collections as we have done in the scalability experiments. Table 10 shows that most of the relevant documents are associated with the collection with medium sized documents. This can be explained by its size, it contains almost half of the full collection.

TABLE 10. Document collection characteristics for documents with different lengths.

Collection Doc. Type	No. of Docs.	No. of Active Queries	Total No. of Unique Relevant Docs.	Avg. No. of Rel. Doc./Query	Median No. of Rel. Doc./Query
Short	139,130	72	1864	27.50	18.5
Medium	193,144	72	3447	52.14	45.0
Long	76,031	72	1612	24.67	21.0

In the experiments, we use the query form Q_M. The graphical representation of *bpref* values in Figure 7 shows that as the document sizes increase the effectiveness in terms of *bpref* values significantly increases ($p < 0.001$) and this is true for all stemming options. We have done no objective analysis of the average number of topics per news articles of the *Milliyet* collection. However, it is our anecdotal observation that in the overwhelming majority of the news articles only one topic is covered. Hence, the persistent increase in effectiveness as the document length increases can be attributed to the fact that longer documents provide better evidence about their contents and hence better discrimination during

retrieval. Our result of having better performance with longer documents (news articles) is consistent with the findings of Savoy (1999, Tables 1a, 1b, A.1). However, note that when document size increases, document representatives could be more precise until a given limit. After this point, we may expect to see the inclusion of more details or non-relevant aspects (under the implicit assumption of a newspaper corpus). Thus longer documents could hurt the retrieval performance in such cases.

In the experiments, for all document length cases, the performance difference of F5 and LV is statistically insignificant. However, the performance difference of these stemmers with respect to NS is statistically significant ($p < 0.001$). These observations reconfirm our previously stated findings for these stemmers, but in different types of documents in terms of their lengths.

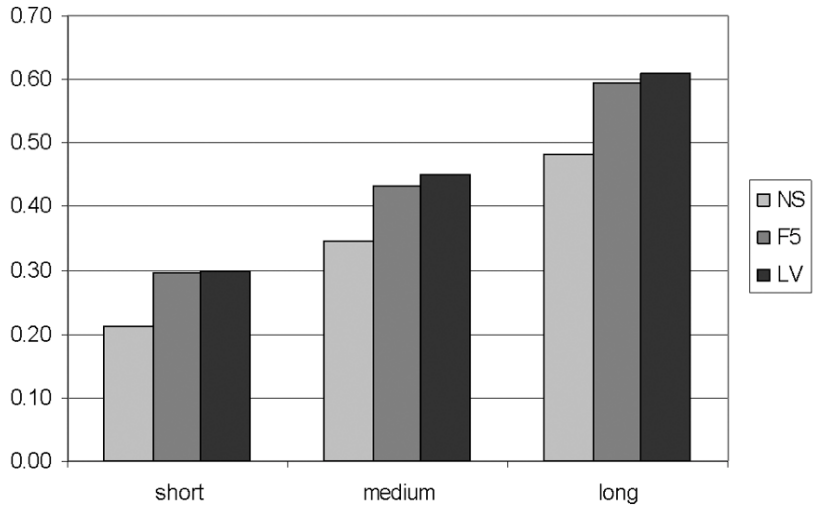

FIG. 7. Query Characteristics and *bpref* values with MF8 using Q_M for different document length collections.

CONCLUSIONS AND FUTURE RESEARCH

In this study we provide the first thorough investigation of information retrieval on Turkish texts using a large-scale test collection. If we revisit our hypotheses stated at the beginning of the article we can list our findings as follows. We show that within the context of Turkish IR and the retrieval model(s) we use in the experiments

- a stopword list has no influence on system effectiveness;
- a simple word truncation approach, a word truncation approach that uses corpus statistics, and an elaborate lemmatizer-based stemmer provide similar performances in terms of effectiveness;
- longer queries improve effectiveness; however, this increase is not linearly proportional to the query lengths;
- longer documents provide higher effectiveness.

Our study conclusively shows that stemming is essential in the implementation of Turkish search engines. With the best performing matching function MF8, the stemming options F5 and LV provide, respectively, 33% and 38% higher performance than that of no stemming.

There are several practical implications of our findings and they are all good news for system developers. No negative impact of not using a stopword list during indexing (with the *tf.idf* model we use in the experiments) has possible desirable consequences, since users may

intentionally submit queries with such common words (Witten, et al. 1999). The use of truncated words in indexing rather than the results of a sophisticated stemmer simplifies the implementation of search engines and improves the system effectiveness with respect to no stemming. Better effectiveness with longer queries is a desirable characteristic since it matches the search engine users' expectations.

In the experiments the matching function MF8 gives a significantly better retrieval performance. Interestingly, MF8 is especially suitable for real life dynamic collections since in this measure the *idf* component can easily be reflected to term weighting on the fly. The *Milliyet* test collection for Turkish, which will be shared with other researchers, is one of the main contributions of this study.

Our work can be enhanced in several ways. The fixed prefix stemming approaches may trim too much, i.e., they may overstem, and the meaning of the stemmed word can be lost. For example, the Turkish word "sinema" (cinema), which is borrowed from English, becomes "sine" with the F4 stemmer. During searching, this stem may return documents related to "sinek" (fly, the insect) since they share the same so called stem "sine." (This is an anecdotal example that we have observed during the experiments.) The other extreme, understemming with long prefix values, has its own problems (Frakes, Baeza-Yates, 1992). The SV and LV stemmers may do a better job in similar situations, but they are imperfect too. It could be possible to find several problematic cases for any stemmer in any language. In real life IR applications, some of the problems introduced by stemming can be resolved before displaying the retrieved documents to the users. For example, in Web applications this can be done during sineppet generation. Furthermore, the stemming process can be improved to handle compound words. Implementation of some other retrieval approaches (like OKAPI (BM25), language modeling (Zobel, Moffat, 2006), and mutual information model (Turney, 2002), *n*-gram-based retrieval (McNamee & Mayfield, 2004), and cluster-based retrieval (Altingovde, Ozcan, Ocalan, Can, & Ulusoy, 2007; Can, Ozkarahan, 1990; Can et al., 2004), all within the context of Turkish IR, are some future research possibilities.

Acknowledgements

We thank our colleagues, friends, and students for their queries. We thank Sengor Altingovde and the anonymous referees for their valuable and constructive comments. This work is partially supported by the Scientific and Technical Research Council of Turkey (TÜBİTAK) under the grant number 106E014. Any opinions, findings and conclusions or recommendations expressed in this article belong to the authors and do not necessarily reflect those of the sponsor.

Appendix

TABLE A.1. Stopword List with 147 words.

ama	böyle	dolayısıyla	her	ki	olmak	sadece	yaptığı
ancak	böylece	edecek	herhangi	kim	olması	şey	yaptığını
arada	bu	eden	herkesin	kimse	olmayan	siz	yaptıkları
ayrıca	buna	ederek	hiç	mı	olmaz	şöyle	yerine
bana	bundan	edilecek	hiçbir	mi	olsa	şu	yine
bazı	bunlar	ediliyor	için	mu	olsun	şunları	yoksa
belki	bunları	edilmesi	ile	mü	olup	tarafından	zaten
ben	bunların	ediyor	ilgili	nasıl	olur	üzere	
beni	bunu	eğer	ise	ne	olursa	var	
benim	bunun	etmesi	işte	neden	oluyor	vardı	
beri	burada	etti	itibaren	nedenle	ona	ve	
bile	çok	ettiği	itibariyle	o*	onlar	veya	
bir	çünkü	ettiğini	kadar	olan	onları	ya	
birçok	da	gibi	karşın	olarak	onların	yani	
biri	daha	göre	kendi	oldu	onu	yapacak	
birkaç	de	halen	kendilerine	olduğu	onun	yapılan	
biz	değil	hangi	kendini	olduğunu	öyle	yapılması	
bize	diğer	hatta	kendisi	olduklarını	oysa	yapıyor	
bizi	diye	hem	kendisine	olmadı	pek	yapmak	
bizim	dolayı	henüz	kendisini	olmadığı	rağmen	yaptı	

* Among letters only “o” is listed as a word (since it is a meaning word in Turkish).

TABLE A.2. News categories.

Category No.	Category	No. of News	Category No.	Category	No. of News
1	Accidents	2	7	Health news	7
2	Acts of violence or war	8	8	Legal/criminal cases	3
3	Art and culture news	7	9	Miscellaneous news	9
4	Celebrity and human interest	2	10	Natural disasters	5
5	Education news	5	11	Political and diplomatic news	5
6	Financial news	10	12	Sports news	9

TABLE A.3. Query topics.

Query No.	Topic (News Category No., No. of Relevant News)	Query No.	Topic (News Category No., No. of Relevant News)
1	kuş gribi (7, 18)	37	Türkiye'de mortgage (6, 95)
2	Kıbrıs sorunu (11, 115)	38	ABD Afganistan savaşı (2, 57)
3	üniversiteye giriş sınavı (5, 131)	39	Yüzüklerin Efendisi-Kralın Dönüşü (3, 41)
4	tsunami (10, 101)	40	beyin göçü (9, 59)
5	mavi akım doğalgaz projesi (6, 68)	41	aile kadın şiddet (2, 74)
6	deprem tedbir önlem (10, 124)	42	sporcuların doping yapması (12, 239)
7	Türkiye PKK çatışmaları (2, 73)	43	ozon tabakasındaki delik (7, 47)
8	film festivalleri (3, 38)	44	Rusya'da okul baskını (2, 99)
9	bedelli askerlik uygulaması (6, 90)	45	İstanbul'da bombalı saldırı (2, 130)
10	stresle başa çıkma yolları (7, 117)	46	Sakıp Sabancı'nın vefatı (4, 101)
11	şampiyonlar ligi (12, 47)	47	Ecevit Sezer çatışması (11, 47)
12	17 ağustos depremi (10, 161)	48	Kıbrıs Türk üniversiteleri (5, 45)
13	Türkiye'de internet kullanımı (9, 220)	49	Türkiye'de 2003 yılında turizm (6, 185)
14	Amerika Irak işgal demokrasi petrol (2, 87)	50	Türkiye'nin nükleer santral çalışmaları (9,53)
15	Türkiye'de futbol şikesi (12,190)	51	hızlı tren kazası (1, 42)
16	Fadıl Akgündüz (8, 26)	52	YÖK'ün üniversitelerimiz üzerindeki etkisi (5, 195)
17	işsizlik sorunu (6, 216)	53	İbrahim Tatlıses'in kadınları (4, 64)
18	2005 F1 Türkiye grand prix (12, 36)	54	parçalanmış aileler (9, 40)
19	ekonomik kriz (6, 69)	55	aile içi şiddet (2, 143)
20	Nuri Bilge Ceylan (3, 35)	56	Türkiye'de kanser (7, 53)
21	Türkiye'de meydana gelen depremler (10, 142)	57	futbol terörü ve holiganizm (12, 200)
22	ABD-Irak savaşı (2, 159)	58	Türkiye'de ikinci el otomobil piyasası (6, 93)
23	Hakan Şükür'ün milli takım kadrosuna alınmaması (12, 41)	59	tarihi eser kaçakçılığı (8, 96)
24	Avrupa Birliği, Türkiye ve insan hakları (11, 98)	60	festival (3, 263)
25	turizm (6, 158)	61	Türkiye'de bayram tatillerinde meydana gelen trafik kazaları (1, 93)
26	Türkiye'deki sokak çocukları (9, 140)	62	öğrenmeyi etkileyen faktörler (5, 123)
27	Türk filmleri ve sineması (3, 151)	63	kekik otu (9, 64)
28	Pakistan depremi (10, 62)	64	telif hakları (3, 51)
29	sanat ödülleri (3, 85)	65	internet ve toplum (9, 201)
30	Avrupa Birliği fonları (6, 75)	66	tarım hayvancılık sorunları (6, 134)
31	futbolda şike (12, 212)	67	İran'da nükleer enerji (11, 237)
32	milletvekili dokunulmazlığı (8, 217)	68	satranç (9, 93)
33	2001 erkekler Avrupa basketbol şampiyonası (12, 35)	69	kalıtsal hastalıklar (7, 65)
34	2002 dünya kupası (12, 27)	70	hiperaktivite ve dikkat eksikliği (7, 36)
35	bilişim eğitimi ve projeleri (5, 156)	71	lenf kanseri (7, 28)
36	global ısınma (9, 128)	72	28 Şubat süreci (11, 76)

References

Ahlgren, P., & Kekalainen, J. (2007). Indexing strategies for Swedish full text retrieval under different user scenarios. Information Processing and Management, 43(1), 81-102.

Altingovde, I. S., Ozcan, R., Ocalan, H. C., Can, F., Ulusoy, O. "Large-scale cluster-based retrieval experiments on Turkish texts." (Poster paper.) In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR '07) (pp. 891-892). Amsterdam: ACM..

Altintas, K., Can, F. (2002) Stemming for Turkish: A comparative evaluation. In Proceedings of the 11th Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN 2002) (pp. 181-188) İstanbul: İstanbul University Press.

Anderson, S., & Cavanagh, J. (2006). Report on the top 200 corporations. Dec.2000, Retrieved October 9, 2006, from http://www.corporations.org/system/top100.html.

Asian, J., Williams, H. E., & Tahaghoghi, S. M. M. (2004). A testbed for Indonesian text retrieval. In Proceedings of the 9th Australasian Document Computing Symposium (ADCS 2004) (pp. 55-58). Melbourne.

Bitirim, Y., Tonta, Y. & Sever, H. (2002). Information retrieval effectiveness of Turkish search engines. Lecture Notes in Computer Science, 2457, 93-103.

Blair, D.C. (2002). The challenge of document retrieval, Part I: Major issues and a framework based on search exhaustivity, determinacy of representation and document collection size. Information Processing and Management, 38(2), 273-291.

Blair, D. C., & Maron, M. E. (1985). An evaluation of retrieval effectiveness for a full-text document-retrieval system. Communications of the ACM, 28(3), 289-299.

Braschler, M., & Peters, C. (2004). Cross-language evaluation forum: Objectives, results, achivements. Information Retrieval, 7, 7-31.

Braschler, M., & Ripplinger, B. (2004). How effective is stemming and decompounding for German text retrieval? Information Retrieval, 7, 291-316.

Buckley, C., & Voorhees, E. M. (2004). Retrieval evaluation with incomplete information. In Proceedings of the 27th International Conference on Research and Development in Information Retrieval (ACM SIGIR '04) (pp. 25-32). Sheffield: ACM.

Cambazoglu, B. B., & Aykanat, C. (2006). Performance of query processing implementations in ranking-based text retrieval systems using inverted indices. Information Processing and Management, 42(4), 875-898.

Can, F., & Ozkarahan, E. A. (1990). Concepts and effectiveness of the cover coefficient-based clustering methodology for text databases. ACM Transactions on Database Systems, 15(4), 483-517.

Can, F., Altingovde, I. S., & Demir, E. (2004). Efficiency and effectiveness of query processing in cluster-based retrieval. Information Systems, 29(8), 697-717.

Can, F., Kocberber, S., Balcik, E., Kaynak, C., Ocalan, H. C., & Vursavas, O. M. (2006). First large-scale information retrieval experiments on Turkish texts. (Poster paper.) In Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR '06) (pp. 627-628). Seattle: ACM.

Can F. (2006). Turkish information retrieval: Past changes future. Lecture Notes in Computer Science, 4243, 13-22.

Carterette, B., Allan, J., & Sitaraman, R. K. (2006). Minimal test collections for retrieval evaluation. In Proceedings of the 29th International Conference on Research and Development in Information Retrieval (ACM SIGIR '06) (pp. 268-275). Seattle: ACM.

CLEF. (2007). Cross language evaluation forum. (http://www.clef-campaign.org/) Accessed on June 24, 2007.

Ekmekcioglu, F.C., & Willett, P. (2000). Effectiveness of stemming for Turkish text retrieval. Program, 34(2), 195-200.

Figuerola, C. G., Gomez, R., Rodriguez, A. F. Z., & Berrocal, J., L. A. (2006). Stemming in Spanish: a first approach to its impact on information retrieval. Working Notes for the CLEF 2001 Workshop 3 September, Darmstadt, Germany, edited by Carol Peters. Retrieved September 3, 2006 from http://www.ercim.org/publication/ws-proceedings/CLEF2/figuerola.pdf.

Fox, C. (1990). A stop list for general text. SIGIR Forum, 24(1-2), 19-35.

Frakes, W. B., & Baeza-Yates, R. (1992). Information Retrieval: Algorithms and Data Structures. Prentice Hall.

Grimes, B. (ed.). (1996). Ethnologue (Summer Institute of Linguistics, 13th edition).

Hafer, M. A., & Weiss, S. F. (1974). Word segmentation by letter successor varieties. Information Storage and Retrieval, 10, 371-385.

Harter, S. P. (1986). Online Information Retrieval: Concepts, Principles and Techniques. San Diego: Academic Press.

Hakkani-Tür, D. Z. (2000). Statistical Language Modeling for Agglutinative Languages. Ph.D. Thesis, Department of Computer Engineering, Bilkent University, Ankara, Turkey.

Harman, D. (1991). How effective is suffixing? Journal of the American Society for Information Science, 42(1), 7-15.

Jansen, B. J., & Spink, A. (2006). How are we searching the World Wide Web? A comparison of nine search engine transaction logs. Information Processing Management, 42(1), 248-263.

Hull, D. (1996). Stemming algorithms: a case study for detailed evaluation. Journal of the American Society for Information Science, 47(1), 70-84.

Kettunen, K., Kunttu, T. & Jarvelin, K. (2005). To stem or lemmatize a highly inflectional language in a probabilistic IR environment? Journal of Documentation, 61(4), 476-496.

Köksal, A. (1981). Tümüyle özdevimli deneysel bir belge dizinleme ve erişim dizgesi: TÜRDER. In the Proceedings of 3. Ulusal Bilişim Kurultayı, 37-44. TBD 3. Ulusal Bilişim Kurultayı, 6-8 Nisan, Ankara, 37-44.

Krovetz, R. (1993). Viewing morphology as an inference process. In Proceedings of the 16th International Conference on Research and Development in Information Retrieval (ACM SIGIR'93) (pp. 191-202). Pittsburgh: ACM.

Larkey, L. S., Ballesteros, L., & Connell, M. (2002). Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis. In Proceedings of the 25th International Conference on Research and Development in Information Retrieval (ACM SIGIR' 02) (pp. 275-282). Tampere: ACM.

Lee, D. L., Chuang, H., & Seamons, K. (1997). Document ranking and the vector-space model. IEEE Software, 14(2), 67–75.

Lewis, G. L. (1988). Turkish grammar, 2nd ed. Oxford University Press, Oxford.

Long, X., & Suel, T. (2003). Optimized query execution in large search engines with global page ordering. In Proceedings of the 29th Very Large Data Bases Conference (VLDB 2004) (pp.129-140). Berlin: Morgan Kaufmann.

McNamee P., Mayfield J. (2004). Character n-gram tokenization for European language text retrieval , Information Retrieval, 7, 73-97.

NTCIR. (2007). NII test collection for IR systems. (http://research.nii.ac.jp/ntcir/). Accessed on June 24, 2007.

Oflazer, K. (1994). Two-level description of Turkish morphology. Literary and Linguistic Computing, 9(2), 137-148.

Popovic, M., & Willett, P. (1992). The effectiveness of stemming for natural-language access to Slovene textual data. Journal of the American Society for Information Science, 43(5), 384-390.

Pembe F. C., & Say ACC (2004). A linguistically motivated information retrieval system for Turkish. Lecture Notes in Computer Science, 3280, 741-750.

Robertson, S. E. (1981). The methodology of information retrieval experiment. In K. Sparck Jones (Ed.), Information retrieval experiment (pp. 9-31). London: Butterworths.

Salton, G., & Buckley, C. (1988). Term weighting approaches in automatic text retrieval. Information Processing and Management, 24(5), 513-523.

Sanderson, M., & Zobel. J. (2005). Information retrieval system evaluation: Effort, sensitivity, and reliability. In Proceedings of the 28th International Conference on Research and Development in Information Retrieval (ACM SIGIR' 05) (pp. 162-169). Salvodor: ACM.

Savoy, J. (1999). A stemming procedure and stopword list for general French corpora. Journal of the American Society for Information Science, 50(10), 944-952.

Savoy, J. (2006). Light stemming approaches for the French, Portuguese, German and Hungarian languages. In Proceedings of the 21st Annual Symposium on Applied Computing (ACM SAC' 06) (pp. 1031-1035). Dijon: ACM.

Semel, T. (2006). The next Yahoo!: Defining the future. Retrieved June 3, 2006 from .http://yhoo.client.shareholder.com/downloads/2006AnalystDay.pdf.

Sever, H., & Bitirim Y. (2003). FindStem: Analysis and evaluation of a Turkish stemming algorithm. Lecture Notes in Computer Science, 2857, 238-251.

Sever, H., & Tonta, Y. (2006). Truncation of content terms for Turkish. CICLing, Mexico (to appear).

Solak, A., & Can, F. (1994). Effects of stemming on Turkish text retrieval. In Proceedings of the Ninth Int. Symp. on Computer and Information Sciences (ISCIS '94) (pp. 49-56). Antalya.

SMART. (2007). SMART FTP Web site. ftp://ftp.cs.cornell.edu/pub/smart/. Accessed on July 2, 2007.

Sparck Jones, K. (1981). Retrieval system tests. In K. Sparck Jones (Ed.), Information retrieval experiment (pp. 213-255). London: Butterworths.

TDT. (2004). TDT 2004 Annotation manual: Version 1.2. August 4, 2004. Retrieved June 25, 2007, from http://projects.ldc.upenn.edu/TDT5/Annotation/TDT2004V1.2.pdf.

TREC. (2007). Text Retrieval Conference. (http://trec.nist.gov). Accessed on June 24, 2007.

Thompson, P, Turtle, H. R., Yang, B., & Flood, J. (1994). TREC-3 ad hoc retrieval and routing experiments using the WIN system. In Proceedings of the Third Text Retrieval Conference, NIST publication #500-226, Gaithersburg (MD), 1994. Retrieved July 1, 2007, from trec.nist.gov/pubs/trec3/papers/west.ps.gz.

Tordai, A., & de Rijke (2006). Four stemmers and a funeral: Stemming in Hungarian at CLEF 2005. Lecture Notes in Computer Science, 4022, 179-186.

Turney, P. D. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In Proceedings of the 40th Annual Meeting of the Assoc. for Computational Linguistics (ACL 2002), (pp. 417-424). Philadelphia.

Voorhees, E. (2005). TREC: Improving information access through evaluation. Bulletin of the American Society for Information Science and Technology, 32(1) October/November 2005. Retrieved October 8, 2006, from http://www.asis.org/Bulletin/Oct-05/voorhees.html.

Voorhees, E. M., & Harman, D. K. (2005). TREC experiments and evaluation in information retrieval. Cambridge, The MIT Press.

Witten, I. H., Moffat, A., & Bell, T. C. (1999). Managing Gigabytes: Compressing and Indexing Documents and Images, 2nd ed. San Francisco, CA, Morgan Kaufmann.

Zobel, J. (1998). How reliable are the results of large-scale information retrieval experiments? In Proceedings of the 21st International Conference on Research and Development in Information Retrieval (ACM SIGIR' 98) (pp. 307-314). Melbourne: ACM.

Zobel, J., & Moffat, A. (2006). Inverted files for text search engines. ACM Computing Survey, 38(2), 1-56.