First Large-Scale Information Retrieval Experiments on Turkish Texts

Fazli Can, Seyit Kocberber, Erman Balcik, Cihan Kaynak, H. Cagdas Ocalan, Onur M. Vursavas
Bilkent Information Retrieval Group
Bilkent University
Bilkent, Ankara 06800, Turkey
canf@cs.bilkent.edu.tr, seyit@bilkent.edu.tr, {ebalcik, kaynak, hocalan, muratv}@ug.bilkent.edu.tr

ABSTRACT
We present the results of the first large-scale Turkish information retrieval experiments performed on a TREC-like test collection. The test bed, which has been created for this study, contains 95.5 million words, 408,305 documents, 72 ad hoc queries and has a size of about 800MB. All documents come from the Turkish newspaper *Milliyet*. We implement and apply simple to sophisticated stemmers and various query-document matching functions and show that truncating words at a prefix length of 5 creates an effective retrieval environment in Turkish. However, a lemmatizer-based stemmer provides significantly better effectiveness over a variety of matching functions.

Categories and Subject Descriptors
H.3.1 [**Content Analysis and Indexing**]: indexing methods; H.3.3 [**Information Search and Retrieval**]: s*earch process, selection process.*

General Terms
Performance, Experimentation.

Keywords
IR Test Collection Creation, Lemmatizer, Stemming, Turkish.

1. INTRODUCTION
In this pioneering work, we present the results of the first large-scale Turkish information retrieval (IR) experiments. The previously published works on Turkish IR are based on test collections of size 500 to 2500 documents, e.g., see [5, 7] (commercial Web search engines do not disclose their methods). We examine the effectiveness of four different simple to sophisticated stemming techniques (no stemming, fixed prefix, language independent successor variety, and a lemmatizer-based stemmer using a morphological analyzer) with eight query-document matching functions.

2. STEMMING
In the first stage we use three stemming options: no stemming (NS), 6-prefix –first six characters- (F6) of each word, and the successor variety (SV) method [3]. Only one of these options, SV, involves "active" stemming. We include F6 since the use of 5-, 6-, or 7-prefix is recommended for rapid and feasible Turkish IR system implementation [6]. The Successor Variety, SV, algorithm determines the root of a word according to the number of distinct succeeding letters for each prefix of the word in a large corpus [3]. The expectation is that the stem of a word would be the prefix at which the maximum successor variety, i.e., the distinct number of successor letters, is observed. Our SV implementation involves adaptations to Turkish and chooses the longest prefix corresponding to the highest SV value.

In the second stage we add several additional prefix options: F3, F4, F5, F7 and a lemmatizer-based stemmer supported by a morphological analyzer for obtaining more accurate stems [1]. A lemmatizer identifies the "lemma" of a word, i.e., word's base form in dictionary. Turkish, an agglutinative language like Finnish and Korean and very different from English, does not have much irregularities. However, for a given word a lemmatizer can provide more than one alternative. In such cases we choose the alternative whose length is closest to the average lemma length (6.58 characters) of word types. If there are multiple candidates we choose the one whose corresponding word type (part of speech, POS, information) is most frequent in Turkish. It is experimentally shown that this approach is more than 90% accurate [1]. In choosing lemmas we also use the length of 5 characters instead of 6.58 (since F5 gives good retrieval results, shown later). By this way we have two lemmatizer-based stemmer versions: LM5 and LM6. For miss spelled and foreign words, which cannot be analyzed by the lemmatizer (about 40% of all distinct words), in an additional **L**M5 version we use the **S**V method for such words, this crossbreed is referred to as LV.

3. EXPERIMENTAL ENVIRONMENT
Our collection contains 408,305 news articles including columns etc. of five complete years, 2001 to 2005, from the Turkish newspaper *Milliyet* (www.milliyet.com.tr). All numbers and words are used after stop word elimination. Our *Milliyet* collection is about 800MB in size and contains about 95.5 million words (tokens) including numbers before stop word elimination. Average article size is 234 tokens. Our stop word list contains 320 words (no numbers) and the number of occurrences of these words corresponds to 15% of all collection tokens. As expected stemming affects document vectors, e.g., NS, F6, and SV result in 150, 134, and 120 average number of unique words (types) per document. The average type lengths for these cases, respectively, are 9.88, 5.66, and 7.23 characters. The posting lists sizes in terms of <term, term frequency pair> are approximately 61, 55, and 49 million entries for NS, F6, and SV.

We use eight query-document matching functions (MF1 to MF8) based on the vector space model. By using the notation of [4] the

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.
ACM 1-59593-369-7/06/0008.

first seven are defined as txc.txx, tfc.nfx, tfc.tfx, tfc.bfx, nfc.nfx. nfc.tfx, nfc.bfx. The first one, MF1: txc.txx, is the well-known cosine function and involves no *idf* component. The next six matching functions, MF2 to MF7, are highly recommended in [4]. Finally, MF8 is a matching function that requires no change in document term weights in dynamic collections by reflecting the *idf* effects of collection changes to query term weights rather than document weights [8, p. 187].

The queries are created according to the TREC ad hoc query tradition by 33 native speakers using binary judgments. The relevant documents are identified by taking the union of the top 100 documents of the 24 possible retrieval combinations, "runs," of the 8 matching functions and the first 3 stemmers NS, F6, and SV. The average pool size and relevant items per query, respectively, are 466 and 104 documents. We have 72 queries after eliminating about 20 queries with too few ($\leq$ 5% of its pool) and too many ($\geq$ 90%) relevant documents since such queries are not good at discriminating retrieval systems. Each query contains about 11 distinct words.

4. EXPERIMENTAL RESULTS

For measuring effectiveness we use the bpref measure to prevent any possible bias effect on the runs not involved in query pool construction [2].

Table I. Bpref values and % improve. of LV wrt SV and F5*

MF	NS	SV	F5	LM5	LV	LV/SV	LV/F5
MF1	.2262	.2851	.2916	.3089	.3154	10.63	8.16
MF2	.3035	.3995	.3843	.3976	.4047	1.30	5.31
MF3	.2949	.3790	.3694	.3872	.3933	3.77	6.47
MF4	.3002	.3900	.3756	.3910	.3976	1.95	5.86
MF5	.2661	.3417	.3533	.3588	.3666	7.29	3.76
MF6	.2819	.3438	.3653	.3595	.3657	6.37	0.11
MF7	.2471	.3199	.3281	.3426	.3495	9.25	6.52
MF8	.3142	.4134	.4146	.4265	.4319	4.48	4.17
C. Av.	.2793	.3591	.3603	.3715	.3781	5.63	5.05

* MF: matching function, NS: bpref value for NS, LV/SV: % bpref improvement of LV with respect to SV.

In order to streamline the final analysis we only include the best representative of the prefix methods to that process. For this purpose we use the 11-point recall precision graphs, an inexact comparison approach but a practical way of solving the choose-one-representative problem. In this comparison the extremes F3 and F7 are definite losers, the others from good to best, in order, are F6, F4, and F5. The difference especially between the last two is insignificant, but F5 is slightly better than F4. In a similar fashion LM5 is slightly better than LM6. In these comparisons we use MF8, the best performer among the matching functions. As explained before we also have LV that takes advantage of LM5 and SV. As a result for the final analysis we have NS, SV, F5, LM5, and LV.

Table I shows that NS is much worse than the others. The most effective one is LV. The SV method and the simple prefix method F5 are also effective, but not as good as LV. The LV stemmer is slightly better than LM5. In order to show that LV is significantly better than SV (i.e., the LV bpref values are significantly greater than that of SV, LV>SV) and F5 (LV>F5) two one-sided matched pair t-tests were performed using the document query matching functions (Table I data). (We use the one-tailed p-values instead of the two tailed since we are trying to show, for example, that LV>SV instead of LV is not equal to SV.) The results showed that the bpref average of LV was significantly greater than that of SV and F5 averages (for both cases $p<.001$). Additional two one-sided matched pair t-tests applied to the MF8 individual query results indicated that the LV values were significantly greater than that of the SV and F5 averages (for both cases $p<.05$). (In the additional tests two queries were eliminated since they were identified as potential outliers.)

5. CONCLUSIONS

We show that truncating words at a prefix length of 5 provides an effective retrieval environment in Turkish. However, a lemmatizer-based stemmer provides significantly better effectiveness over a variety of matching functions. In the experiments the matching function MF8 gives a significantly better retrieval performance. Interestingly, MF8 is especially suitable for real life dynamic collections since it reflects the *idf* component of query-document matching to query term weights and therefore requires no re-weighting of document terms. The TREC-sized test collection for Turkish, which will be shared with other researchers, is one of the main contributions of this study. Our findings about the stemming approaches can be reflected to some other agglutinative languages. Currently we are working on further experiments.

6. ACKNOWLEDGMENTS

We thank the anonymous referees, Ben Carterette, Jon M. Patton, and the query participants - our colleagues, friends, and students.

7. REFERENCES

[1] Altintas, K., Can, F., Patton, J. M. Language change quantification using time-separated parallel translations. *Literary and Linguistic Computing* (resubmitted after rev.).

[2] Buckley, C., Voorhees, E. M. Retrieval evaluation with incomplete information. ACM SIGIR Conf, pp. 25-32, 2004.

[3] Hafer, M. A., Weiss, S. F. Word segmentation by letter successor varieties. *Infor. Stor. Retr.* 10, 371-385, 1974.

[4] Salton, G., Buckley, C. Term weighting approaches in automatic text retrieval. *IPM 24*, 513-523, 1988.

[5] Sever, H., Bitirim Y. FindStem: analysis and evaluation of a Turkish stemming algorithm. *LNCS* 2857: 238-251, 2003.

[6] Sever, H., Tonta, Y. Truncation of content terms for Turkish. *CICLing Feb. 2006, Mexico* (to appear).

[7] Solak, A., Can, F., Effects of stemming on Turkish text retrieval. ISCIS Conf., pp. 49-56, 1994.

[8] Witten, I. H., Moffat, A., Bell, T. C. *Managing Gigabytes: Compressing and Indexing Documents and Images, 2nd ed.* San Francisco, CA, Morgan Kaufmann, 1999.