ANADOLU ÜNİVERSİTESİ BİLİM VE TEKNOLOJİ DERGİSİ –A
Uygulamalı Bilimler ve Mühendislik

ANADOLU UNIVERSITY JOURNAL OF SCIENCE AND TECHNOLOGY –A
Applied Sciences and Engineering
Cilt/Vol.:11-Sayı/No: 2 : 163-172 (2010)

A LANGUAGE MODELING APPROACH TO TURKISH TEXT RETRIEVAL

Özgür YILMAZEL [1]

ABSTRACT

We used Lemur Toolkit, an open source toolkit designed for Information Retrieval research, for our automated indexing and retrieval experiments on a TREC-like test collection for Turkish language. We investigate effectiveness of three retrieval models Lemur supports, especially Language modeling approach to Information Retrieval, combined with language specific preprocessing techniques. Our experiments show that language specific preprocessing significantly improves retrieval performance for all retrieval models. Also Language Modeling approach is the best performing retrieval model when language specific preprocessing applied.

Keywords: Turkish information retrieval; Lemur toolkit; Language modeling.

TÜRKÇE METİN GERİ GETİRİMİNDE DİL MODELLEME YAKLAŞIMI

ÖZ

Bu çalışmada, bilgi erişimi araştırması için tasarlanmış açık kaynak kodlu bir araç olan Lemur kullanılarak, Türkçe dili için hazırlanmış TREC benzeri bir derlem üzerinde otomatik indeksleme ve geri getirme deneyleri gerçekleştirildi. Bilgi erişiminde dil modelleme yaklaşımı başta olmak üzere Lemur tarafından desteklenen üç geri getirme modeli ve dile özgü ön işleme teknikleri araştırıldı. Deneylerimiz, dile özgü ön işleme tekniklerinin tüm geri getirim modelleri için geri getirme performansını artırdığını gösterdi. Ayrıca Türkçe dili için en iyi performans dil modelleme yaklaşımından elde edildi.

Anahtar Kelimeler: Türkçe bilgi erişimi, Lemur aracı, Dil modelleme.

[1,] Anadolu University, Department of Computer Engineering, 2 Eylul Kampusu, Eskisehir, Turkey.
Phone: +90 – 533 336 3199 Fax: +90 – 222 – 323 0999 E-mail: ozgur@anadolu.edu.tr

Recieved: 29 October 2010; **Revised:** 30 November 2010; **Revised:** 20 December 2010; **Accepted:** 22 December 2010

1. INTRODUCTION

Information retrieval is a broad science with expanding subfields, while most new approaches to Information Retrieval (IR) field are first introduced using English document collections, IR experiments in other languages also move parallel to current trends.

There are various studies on Turkish Information Retrieval research incorporating widely used vector space and probabilistic retrieval models combined with different language specific preprocessing. However, there is no published study comparing effectiveness of language modeling approach for Turkish text retrieval.

In this paper we compare retrieval performance of Turkish as an agglutinative language with productive inflectional and derivational suffixations, both from language modeling approach and conventional retrieval approaches. For this purpose three main retrieval models were used in our experiments. They are Lemur TF-IDF, OKAPI and Language Modeling. Also we investigate how language specific properties of Turkish affect retrieval performance.

Our experiments are based on a TREC-like test collection consisting of 485.000 documents, 72 ad-hoc queries and relevance judgments. We made experiments with different settings to find optimum parameters and also tried to do language specific improvements for all three retrieval models. We used Lemur Toolkit[1], an open source toolkit designed for Information Retrieval research, for our automated indexing and retrieval experiments.

The organization of this paper is as follows. Section 2 gives a quick reminder of retrieval models related to the context of this paper and briefly summarizes the previous work on Turkish Information Retrieval. Section 3, focuses on retrieval methods Lemur Toolkit supports and gives some information about toolkit. Section 4 presents the details of our experimental setup including dataset and stemming algorithms that we used in this paper, Section 5 contains the experimental results, and Section 6 provides concluding remarks and future work.

1. http://www.lemurproject.org/
2. http://trec.nist.gov

2. PREVIOUS WORK

Since the first Text Retrieval Conference (TREC[2]) (Harman, 1993), different researches were conducted on large text collections and different retrieval models were proposed. Vector space (Salton, et al., 1975) model represents documents and queries as high dimensional vectors. Documents for a given query are ranked according to similarity between document and query vectors. K. S. Jones (Jones, 1988) showed that using inverse document frequency and term frequency together as a term weighing method is much better than using term frequency alone. Also, OKAPI system (Walker et al., 1988), a probabilistic model, evolved at the TREC conferences, integrates document length normalization factor into term weighing methods.

In addition to these conventional models, J. Ponte and W. B. Croft (Ponte & Croft, 1998) proposed a model that scores documents according to the probability of query generated by document language model. Zhai and Lafferty (et., 2001) examined effects of different smoothing parameters on language modeling based IR performance.

Turkish IR research also progresses. F. Can and Bilkent Information Retrieval Group (Can, et al., 2008) created the first large-scale TREC like Turkish IR text collection (Milliyet Collection). Also they compared retrieval performance of different stemming approaches and term weighing strategies using this collection. Later they performed cluster-based retrieval (CBR) experiments on the same test collection (Altingovdeet al., 2007). We also use the same text collection (Milliyet Collection) for experiments in this paper.

A. Arslan and Ö. Yılmazel (Arslan and Yilmazel, 2008) compare Turkish text retrieval performances of relational databases versus open source retrieval library Lucene using the same test collection.

3. LEMUR TOOLKIT

Lemur Toolkit is an open source IR research system developed collaboratively by University of Massachusetts, Amherst and Carnegie Mellon University. It is written in C and C++ languages and it is available with C# and JAVA APIs. It works under UNIX and Windows based operating systems.

3.1 Indexing

Lemur has two types of inverted indexes: KeyfileIncIndex and IndriIndex.[3] We used Indri type index in our experiments to benefit from its built-in support for UTF-8 encoded documents, and also because of support files created for language modeling algorithm.

3.2 Retrieval

Experiments on this paper are based on three retrieval methods. Main retrieval model is a unigram language-modeling algorithm that ranks documents by similarity of document and query language models using Kullback-Leibler (Cover and Thomas, 1991) divergence as a measure. Other two retrieval models are OKAPI retrieval algorithm (Robertsonet et al., 1992) (Walker, et al.) and a dot product function (Zhai) using a TF-IDF variant for term weighing.

3.2.1 Lemur TF-IDF Model

In lemur TF-IDF model each document ($\vec{d}$) and query ($\vec{q}$) are represented as vectors. Here, tf is term frequency and $idf(t_n)$ is inverse document frequency where n is the number of terms in vocabulary.

$$\vec{d} = (tf_d(x_1)idf(t_1), tf_d(x_2)idf(t_2), ...tf_d(x_n)idf(t_n))$$

$$\vec{q} = (tf_q(y_1)idf(t_1), tf_q(y_2)idf(t_2), ...tf_q(y_n)idf(t_n))$$

Detailed formulas of tf_d, tf_q and $idf(t_n)$ can be found at (Zhai). Similarity of two vectors is calculated using following vector dot product.

$$Sim(\vec{d}, \vec{q}) = \sum_{i=1}^{n} tf_d(x_i)tf_q(y_i)idf(t_i)^2 \qquad \textbf{(1)}$$

3.2.2 OKAPI Model

OKAPI model is a member of probabilistic retrieval models. Documents are ranked according to decreasing probability of their relevance to a query. Okapi system evolved to BM25 weighing formula at TREC-3 (Robertson et al., 1992) that is a mixture of BM11 and BM15 formulas. If there is no present relevance information the scoring function $score(d,q)$ is defined as the following:

Given following notations, number of documents N, document frequency of a term df_t, scaling constants k_1 and k_3, document length normalization constant b, document length L_d, average document length L_{ave}, document d, query q, term frequency of a term t in document d and query q respectively tf_{td} and tf_{tq}, scoring is defined as the following formula (Walker, et al.).

$$score(d,q) = \Sigma_{t \in q}\left[\log\frac{N}{df_t}\right] \cdot \frac{(k_1+1)tf_{td}}{k_1((1-b)+b\times(L_d/L_{ave}))+tf_{td}} \cdot \frac{(k_3+1)tf_{tq}}{k_3+tf_{tq}}$$

(2)

3.2.3 Language Modeling

Language modeling is a statistical estimation to predict the probability of a word sequence based on the probability distribution generated from some language sample. From IR perspective documents are ranked according to the probability of query Q generated by a document language model.

In Lemur instead of calculating the probability of generation of query Q, language models are generated for both query and documents. Documents are ranked according to how close the document and query language models are to each other. Language modeling approach uses Kullback-Leibler (KL) divergence (Cover and Thomas, 1991) to calculate the distance between two language models, so negative of the KL divergence is used for ranking documents (Andet al., 2002).

Let M_D, M_Q and M_C be language models for document D, query Q and collection C respectively.

$p_s(w|M_D)$: Smoothed probability of word w in M_D, given w is seen in document D

α_D: Document dependent constant.

$p(w|\mathrm{M_C})$ $p(w|\mathrm{M_Q})$: Probability of word w in M_C and M_Q.

3. http://www.lemurproject.org/lemur/indexingfaq.php

$p_{ml}(w \mid M_D)$: Probability of word w in M_D using maximum likelihood estimate.

$tf_{w,d}$: Count of word w in document d.

$tf_{u,d}$: Number of unique terms in document d.

Final equation (And, et al., 2002) is given as negative of the divergence (Cover & Thomas, 1991) of query and document language models:

$$-D(M_Q||\mathrm{M_D}) = \sum_{w \in D \cap Q} p(w|\mathrm{M_Q}) \log \frac{p_S(w|\mathrm{M_D})}{\alpha_D p(w|\mathrm{M_C})} + \log \alpha_D \quad (3)$$

Documents content is very sparse for creating a perfect language model. In order to avoid from underestimating the probability of words that do not exist in documents, a smoothed probability of $p_S(w \mid M_D)$ is used. Three interpolation based smoothing methods (Zhai and Lafferty, 2001) are available at Lemur[4] for calculating $p_S(w \mid M_D)$ and α_D, given in Table 1. There also exists back off based methods but we didn't do experiments with them.

4. EXPERIMENTAL SETUP

In this section, firstly, we describe test collection that we used in our study, and then we give information about Turkish stemming algorithms implemented and used with Lemur Toolkit[4]. Finally we talk about parsing decisions we made for our experimental runs.

4.1 Test Collection

In this study Milliyet Collection (Can, et al., 2008) created by F. Can and Bilkent Information Retrieval Group was used. Milliyet Collection consists of 408.305 documents and is approximately 800 MB in size; documents in the collection are news articles from Turkish newspaper Milliyet[5].

4.1.1 Documents

Each document in collection is in XML format consisting of one root node {DOC} and eight child nodes {DOCNO, SOURCE, URL, DATE, TIME, AUTHOR, HEADLINE, TEXT}. An example of a Milliyet Collection document is shown in Figure 1. The HEADLINE and TEXT fields contain searchable textual information; we merged content of those two fields into a single field.

4.1.2 Queries

Queries created for test collection consists of three parts (title, description, narratives) from shortest to longest one, which are similar to a typical TREC query. Title field consist a few words that best describe the topic. The description field is the description of the topic in one or two sentences. The narrative gives a more explanation about the topic. In the TREC terminology, each test information need is referred as a topic. An example of an information need in Milliyet Collection is illustrated in Figure 2. We merged title and description parts into a single query; they are named as middle length queries in our experiments. Linguistic preprocessing methods were applied to both queries and documents identically.

4.1.3 Relevance Judgments

Relevance judgments are answer keys marking the relevant documents, F. Can states that they used pooling concept (Can, et al., 2008) while creating relevance judgments so that relevance judgments are incomplete. It is impossible to create fully judged right answers for large sized collections. For example, assessors would have to judge total number of 408.305x72= 29,397,960 document-query pairs in this case.

4.2 Stemming Algorithms for Turkish

For grammatical reasons, documents use different variations of a word, such as kitap, kitaplar, kitapta and kitabım (Manning et al., 2008). These different variations prevent a string match between a query and a document. Additionally, there are families of derivationally related words with similar meanings, such as demokrasi, demoratik, and demokratikleşme (Manning et al., 2008). In many situations, it would be useful for a search for one of these words to return documents that contain another word in the set, since a user who runs a query on "üniversite" would probably also be interested in documents that contain the word "üniversiteler".

4. http://www.lemurproject.org/lemur/retrieval.php
5. http://www.milliyet.com.tr

In Turkish new words are formed from their root using derivational affixes, words mostly take both derivational and inflectional affixes. Stemming is the process of removing inflectional suffixes (or sometimes derivational suffixes) from words in order to reduce them to a common base form. Conventional retrieval methods studied on Turkish IR show great performance improvements when stemming applied. We also had similar results in our experiments.

4.2.1 No Stemming

While indexing, only default parser explained in section 4.3 used, no stemming applied. As it was mentioned in other two stemming methods, stemming method was applied after default parsing operation.

Table1. List of smoothing methods available at Lemur Toolkit

Method	$p_s(w \mid M_D)$	α_D	*Parameter*
Jelinek-Mercer	$(1-\lambda)\,p_{ml}(w \mid M_D) + \lambda\, p(w \mid M_C)$	λ	λ
Dirichlet	$\dfrac{tf_{w,d} + \mu\, p(w \mid M_C)}{\sum_w tf_{w,d} + \mu}$	$\dfrac{\mu}{\sum_w tf_{w,d} + \mu}$	μ
Absolute Discounting	$\dfrac{\max(tf_{w,d} - \delta, 0)}{\sum_w tf_{w,d}} + \dfrac{\delta\, tf_{u,d}}{\sum_w tf_{w,d}}$	$\dfrac{\delta\, tf_{u,d}}{\sum_w tf_{w,d}}$	δ

```
- <DOC>
    <DOCNO>63102</DOCNO>
    <SOURCE>Milliyet v.01</SOURCE>
    <URL>www.milliyet.com.tr/2002/01/19/yazar/civaoglu.html</URL>
    <DATE>2002/01/19/</DATE>
    <TIME />
    <AUTHOR>Güneri CIVAOĞLU</AUTHOR>
    <HEADLINE>IMF'yi hortumlamak...</HEADLINE>
    <TEXT>Wall Street Journal'da çıkan şu satırların altı çizilerek
      düşünülmelidir "Arjantin'de hurdaya dönen ekonomi treni tüyler
      ürpertici haldeyse, o zaman ABD'nin neden en hayati müttefiki
      Türkiye'yi aynı raya ittiği sorgulanmalıdır." ...</TEXT>
  </DOC>
```

Figure 1. Document 63102 in Milliyet collection

```
- <top>
    <QueryID>298</QueryID>
    <Title>Ekonomik kriz</Title>
    <Description>Türkiye'de ekonomik krize neden
      olan olaylar.</Description>
    <Narrative>Türkiye'de son bir kaç yıl içinde olan
      ekonomik krizlerin nedenleri ve bunlara zemin
      hazırlayan olaylar.</Narrative>
  </top>
```

Figure 2. Topic 298 in Milliyet collection

4.2.2 Snowball Based Stemmer

A stemmer developed by Evren (Kapusuz) Çilden using Snowball[6] string processing language. Turkish words are analyzed with an affix stripping approach without any dictionary lookups (Eryigit and Adali, 2004).

4.2.3 Zemberek Stemmer

Zemberek[7] is an open source natural language processing library designed for Turkic languages especially for Turkish. It provides root forms of given words using a root dictionary-based parser combined with Natural Language Processing algorithms. It handles special cases for suffixes and can be used as a lemmatizer based stemmer. Our stemmer incorporating Zemberek removes the inflectional suffixes from terms.

4.3 Parsing Decisions

For parsing we used our own parser; after parsing we applied chosen stemming method and fed all terms into Lemur index directly using ParsedDocument[8] structure. Our parser splits terms by white space, folds them to lowercase, removes preceding and trailing non-alphanumeric characters. Stop word removal was applied before and after stemming operation if enabled. Our stop word list has 148 words for Turkish.

5. EXPERIMENTAL RESULTS

Evaluation standards of an IR system evolve around relevance. Documents are judged either as relevant or nonrelevant to given information needs. For ranked retrieval results we use two mostly used metric MAP and *bpref* (Buckley and Voorhees, 2004). C. Buckley and E. M. Voorhees introduced *bpref* in SIGIR 2004 and stated that it is more robust than MAP to incomplete relevance judgments. Since our collection has incomplete relevance judgments, we compared different retrieval methods with different stemming options using *bpref* values. Eleven point precision-recall graphs of different retrieval methods are also given on the same figure for evaluating their performance. For evaluation of retrieval results we used a java-based standard trec_eval[9] application included in Lemur.

6. http://snowball.tartarus.org/
7. http://code.google.com/p/zemberek/
8. http://ciir.cs.umass.edu/~strohman/indri/
9. http://trec.nist.gov/trec_eval

In Table 2 we compare effectiveness of stop words to retrieval performance. We point to stop word effectiveness using three retrieval algorithms (Lemur TF-IDF, OKAPI, Language modeling). No stemming applied and queries used in our stop word effectiveness tests are middle length (merged topic and description parts of information needs).

Stop words don't have great effects on retrieval performance when we look at *bpref* values of different retrieval methods; we see the smallest change in Lemur TF-IDF model, which is most strict in *idf* strategy given in Equation 1 with double *idf* values. Inverse document frequency (*idf*) helps eliminating stop words effects on retrieval results. OKAPI and language modeling results also don't have significant changes but differences in their *bpref* values are greater than Lemur TF-IDF model. OKAPI formula in Equation 2 also has single *idf*, the denominator in language modeling formula in Equation 3. Also creates an effect similar to *idf* as mentioned in the study of Zhai and Lafferty (Zhai and Lafferty, 2001).

Our evaluations using Lemur TF-IDF model show that in no stemming applied tests we get best *bpref* values using parameters $k_1=1$, $k_3=1000$, $b=0.2$. When Snowball based and Zemberek stemmers applied we get best *bpref* values using parameters $k_1=1$, $k_3=1000$, $b=0.4$. Test runs with stemming have higher b values than without stemming runs. That means more document length normalization needed, due to the fact that stemming operation decreased count of unique words in the collection and led to higher term frequencies as it is in long documents. Best *bpref* values of our runs with Lemur TF-IDF model are given in Table 3.

When we study with OKAPI model we get best results with Zemberek stemmer. In no stemming applied tests we get best *bpref* values using parameters $k_1=1.4$, $k_3=1000$, $b=0.1$. Runs with Snowball and Zemberek stemmers applied we get best *bpref* values using parameters $k_1=1$, $k_3=1000$, $b=0.75$. Same as in Lemur TF-IDF model when stemming applied increasing document length normalization constant b gives better results. Table 4 shows best *bpref* values of our runs with OKAPI model.

Stemming improves retrieval performance in Turkish as an agglutinative language. Eleven-point precision-recall graph in Figure 3 shows the best performances of IR models we studied.

While OKAPI performs best at low (0-0.2) and middle (0.2-0.8) recall, Lemur TF-IDF and OKAPI performs best at high (0.8-0.2) recall. Language Modeling with Jelinek-Mercer smoothing is the worst performer and Bayesian smoothing using Dirichlet priors is the best performing smoothing method.

We have also made tests with Lemur TF-IDF model using other term weighing approaches, which are Raw TF and Log TF. Other weighing strategies do not perform well compared to default OKAPI derived term weighing. Log TF is better than raw TF since it normalizes term frequency values. In Table 5 we present the best *bpref* values of our runs using different term weighing approaches and stemming options.

In language modeling approach we compared *bpref* values of three smoothing methods with three stemming options using interpolation based smoothing strategy.

Table 2. Best *bpref* values with (SW+) and with out (SW-) stop word removal applied.

	Lemur TF-IDF	OKAPI	Jelinek-Mercer	Dirichlet	Absolute Discounting
SW-	0.4376	0.4370	0.4048	0.4292	0.4158
SW+	0.4324	0.4230	0.3919	0.4204	0.4006

Table 3. Best *bpref* values of Lemur TF-IDF model with three stemming options and stop word elimination (SW+).

No Stemming SW+	Snowball SW+	Zemberek SW+
0.4324	0.5130	0.5096

Table 4. Best *bpref* values of OKAPI model with three stemming options and stop word elimination (SW+).

No Stemming SW+	Snowball SW+	Zemberek SW+
0.4230	0.5068	0.5138

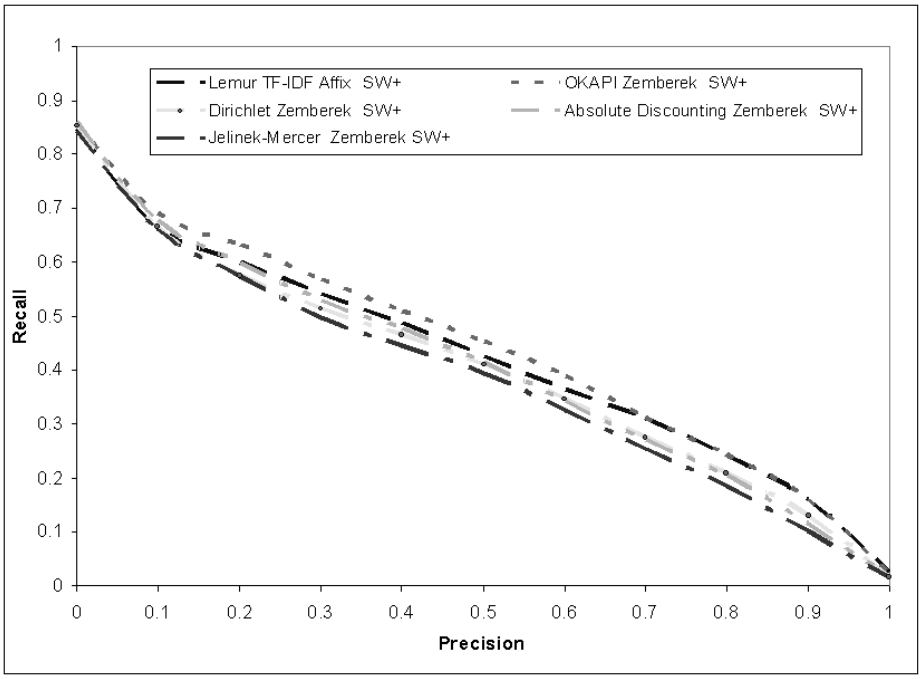

Figure 3. Eleven point precision-recall graph of three IR models with best *bpref* results in all runs.

Table 5. Best *bpref* values of three term weighing approach (OKAPI TF, LOG TF, and RAW TF)

	No Stemming SW+	Snowball SW+	Zemberek SW+
OKAPI TF	0.4324	0.5130	0.5068
LOG TF	0.4084	0.4547	0.4523
RAW TF	0.3397	0.3547	0.3432

Bayesian smoothing using Dirichlet priors was the best performing method, absolute discounting was the second and Jelinek-Mercer was the worst performing in our test runs with optimum parameter values. When the best results are taken as measure Bayesian smoothing has best *bpref* values around prior value of 2000 without stemming and around prior value of 1000 with stemming. Absolute discounting has best *bpref* values around delta value of 0.8 without stemming and around 0.7 with stemming. Conversely, Jelinek-Mercer has best *bpref* values around delta value of 0.3 without stemming and around 0.5 with stemming. The relation between smoothing parameters and stemming based on best *bpref* values is given in Table 6.

The same as in OKAPI and Lemur TF-IDF models, stemmed documents need higher document length normalization values than non-stemmed ones. This is also observed in language modeling. Smaller α_D, in Bayesian smoothing and absolute discounting models, means penalizing longer documents more. Optimum prior (μ) and delta (δ) values of stemming applied tests have smaller α_D values corresponding to more penalization of longer documents. Jelinek-Mercer method does not have a document dependent term α_D for tuning document length effect, so when stemming applied in order to get better results we have to penalize documents more by increasing λ value, the same we did with two other smoothing methods by decreasing prior and delta values. Although smoothing parameters don't increase or decrease consistently between stemmed versions, it is clear that both stemmed versions need more document normalization than non-stemmed ones.

We see that language modeling using Bayesian smoothing is the best performing among three smoothing methods. A comparison of three retrieval models based on best *bpref* values is given in Table 7. As C. Zhai and J. Lafferty (Zhai and Lafferty, 2001) explained in their paper, smoothing of the document language model shows some similarities with traditional heuristics, such as TF-IDF weighing and document length normalization. The same similarity they mentioned is seen in our studies too. In addition to their implications, the effect of stemming in Turkish Information Retrieval is similar in two conventional retrieval models and language modeling approach.

6. CONCLUSIONS AND FUTURE WORK

IR models we used in our experiments meet at the same idea directly or indirectly, the importance of term frequency, inverse document frequency, stemming and document length normalization. We investigated effects of these key concepts in our experiments, especially for language modeling on Turkish IR.

All three IR models have similar responses to the properties mentioned above. Stemming applied experiments in all models give up to 20% performance improvements. Our results are clear evidence of the importance of stemming on Turkish Information Retrieval and are also a clue for other agglutinative languages. A lemmatizer-based stemmer (Zemberek) using morphological rules gives best results. Language modeling is the best performing retrieval model with Zemberek stemmer. When no special linguistic preprocessing applied Lemur TF-IDF is the best performing retrieval model.

Behavior of language modeling is also similar to other two models (Lemur TF-IDF, OKAPI). Also, as shown in Table 7, while Dirichlet is the third best by *bpref* values in no stemming run, it is the first with a lemmatizer-based stemmer (Zemberek). This might be a sign of language specific dependency of language modeling framework.

We will continue to evaluate the effect of different smoothing methods for language modeling framework on Turkish text retrieval, also if available on different Turkish test collections.

Table 6. Best *bpref* values and optimum parameters of three smoothing method with three stemming option.

	No Stemming SW+	Snowball SW+	Zemberek SW+
Jelinek-Mercer	0.3933 ($\lambda = 0.3$)	0.4808 ($\lambda = 0.5$)	0.4842 ($\lambda = 0.4$)
Dirichlet	0.4206 ($\mu = 2000$)	0.5063 ($\mu = 1000$)	0.5148 ($\mu = 500$)
Absolute Discounting	0.4007 ($\delta = 0.75$)	0.4869 ($\delta = 0.7$)	0.4916 ($\delta = 0.7$)

Table 7. Best *bpref* values of three retrieval methods with three stemming options.

	No Stemming SW+	Snowball SW+	Zemberek SW+
Lemur TF-IDF	**0.4324**	**0.5130**	0.5096
OKAPI	0.4230	0.5068	0.5138
Jelinek-Mercer	0.3933	0.4808	0.4842
Dirichlet	0.4206	0.5063	**0.5148**
Absolute Discounting	0.4007	0.4869	0.4916

REFERENCES

Altingovde, I.S., Ozcan, R., Ocalan, H.C., Can, F. and Ulusoy, O. (2007). *Large-scale cluster-based retrieval experiments on Turkish texts*. Paper presented at the Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval.

And, P.O., Ogilvie, P. and Callan, J. (2002). *Experiments Using the Lemur Toolkit.* Paper presented at the in Proceedings of the Tenth Text Retrieval Conference (TREC-10).

Arslan, A. and Yilmazel, O. (2008). 19-22 Oct. 2008). *A comparison of Relational Databases and information retrieval libraries on Turkish text retrieval.* Paper presented at the Natural Language Processing and Knowledge Engineering, 2008. NLP-KE '08. International Conference on.

Buckley, C. and Voorhees, E.M. (2004). *Retrieval evaluation with incomplete information*. Paper presented at the Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval.

Can, F., Kocberber, S., Balcik, E., Kaynak, C., Ocalan, H.C., and Vursavas, O.M. (2008). Information retrieval on Turkish texts. *J. Am. Soc. Inf. Sci. Technol. 59*(3), 407-421.

Cover, T.M. and Thomas, J.A. (1991). *Elemets of information theory*. Wiley-Interscience.

Eryigit, G. and Adali, E. (2004). *An Affix Stripping Morphological Analyzer For Turkish.* Paper presented at the IASTED International Conference on Artificial Intelligence and Applications, Innsbruck, Austria.

Harman, D. (1993). *Overview of the first TREC conference*. Paper presented at the Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval.

Jones, K.S. (1988). A statistical interpretation of term specificity and its application in retrieval *Document retrieval systems* (pp. 132-142): Taylor Graham Publishing.

Manning, C., Raghavan, P. and Schutze, H. (2008). *Introduction to Information Retrieval*: Cambridge University Press.

Ponte, J.M. and Croft, W.B. (1998). *A language modeling approach to information retrieval*. Paper presented at the Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval.

Robertson, S., Walker, S., Hancock-Beaulieu, M., Gull, A. and Lau, M. (1992). *Okapi at TREC-3.* Paper presented at the Text REtrieval Conference.

Salton, G., Wong, A. and Yang, C.S. (1975). A vector space model for automatic indexing. *Commun. ACM 18*(11), 613-620.

Walker, S., Robertson, S.E., Boughanem, M., Jones, G.J.F. and Jones, S. (1998). Okapi at TREC-6 - Automatic ad hoc, VLC, routing, filtering and QSDR.

Zhai, C. Notes on the Lemur TFIDF model, from http://www.cs.cmu.edu/~lemur/1.0/tfidf.ps

Zhai, C. and Lafferty, J. (2001). *A study of smoothing methods for language models applied to Ad Hoc information retrieval*. Paper presented at the Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval.