Turk J Elec Eng & Comp Sci, Vol.20, No.5, 2012, © TÜBİTAK
doi:10.3906/elk-1010-819

Effects of diacritics on Turkish information retrieval

Adil ALPKOÇAK*, Meltem CEYLAN
Department of Computer Engineering, Dokuz Eylül University, Tınaztepe Campus, 35160 Buca, İzmir-TURKEY
e-mail: alpkocak@cs.deu.edu.tr

Received: 04.10.2010

Abstract

We investigate the effects of improper use of diacritics in the Turkish alphabet on information retrieval. A diacritic is simply a supplementary sign added to a letter to change the sound value of the letter, and the Turkish alphabet has 5 special letters derived from Latin by adding different diacritics. The statistical analysis performed in this study shows that retrieval performance significantly decreases when documents and queries contain letters with different forms, such that documents consist of letters with diacritics while queries consist of standard Latin letters and vice versa. In order to tackle this challenge, we propose 3 approaches: token normalization by equivalence classes, document expansion, and query expansion. The experimental evaluations carried on the Bilkent Turkish information retrieval test collection suggests that the proposed approaches are promising as a remedy in this line of research.

Key Words: *Turkish information retrieval, diacritics, document expansion, query expansion*

1. Introduction

Although English seems to be the most popular language for Internet searches, non-English languages have great importance all over the Internet. The growth of the ratio of non-English languages online leads to new information retrieval problems depending on the characteristics of these languages. Language-dependent problems are not limited to the encodings of systems, but also involve the compatibility and utility of these systems. With the development processes of encoding systems, all characteristics of a language become available in the digital world. The information retrieval community deals with developing new methods to improve the performance of systems working on these multilingual document collections. However, each language uses different alphabets; thus, different language characteristics require more intelligent and customized search engines to overcome such language-specific problems and, hence, more language-specific methods are required to improve these results. New research is emerging on this subject as the rate of non-English web pages grows drastically.

Multialphabet text data obviously leads to miswriting or typing errors in both documents and queries. The Turkish alphabet is a Latin-based alphabet used for writing the Turkish language and consists of 29 letters, a certain number of which (Ç, Ğ, I, İ, Ö, Ş, and Ü) were derived from the Latin alphabet with the addition of

*Corresponding author: Department of Computer Engineering, Dokuz Eylül University, Tınaztepe Campus, 35160 Buca, İzmir-TURKEY

4 diacritical marks for the phonetic requirements of the language. In Turkish, typing errors most often occur with these special Turkish letters with diacritics. Users may prefer to type Turkish text without Turkish letters, whether for reasons of speed, laziness, avoidance of encoding errors, or habits born of the days when Turkish letters did not exist on many computer systems. However, improper typing of Turkish letters may also lead to a loss in meaning. As human readers, we can mostly recover this meaning loss from the context automatically. However, in contrast, information retrieval systems are not smart enough to close this gap in meaning. As a result, the performance measure of the retrieval process decreases because the retrieval system does not match the exact terms in the index. From the point of view of the Turkish alphabet, we require more research on what we do miss out on with documents or queries that are not properly typed with the Turkish alphabet.

In the literature, there are a limited number of studies dealing with this problem. Bar-Ilan and Gutman studied the ways in which search engines handle non-English queries [1]. Grefenstette and Nioche calculated the growth of a number of European languages on the Internet [2]. Soraka detailed the performance of search engines for Polish, and Choros tested the effectiveness of retrieval queries while using Polish words with diacritics [3,4]. Similar to the work of Soraka, Moukdad studied retrieval queries for Arabic [5]. Additionally, Daoud dealt with morphological analysis and diacritical Arabic text compression [6]. For Turkish, Bitirim et al. inspected the information retrieval effectiveness of Turkish search engines for 17 specific queries, where 2 of the queries included typographical errors in the Turkish letters [7]. According to their study, the retrieval efficiency of a query changed if the Turkish letters of the query were written with the English alphabet. The number of relevant documents decreased with the use of Turkish letters in a query. They reported that Turkish letters such as i, ş, and ç in queries leads to problems for Turkish search engines.

To the best of our knowledge, although many different information retrieval studies have been done for the Turkish language, the performance loss caused by the improper use of Turkish letters with diacritics has not been studied. Hence, this study focuses on the effects of Turkish letters in documents and queries with respect to information retrieval. We first clarify the problem of the performance loss caused by misuse of diacriticized letters in Turkish text. We then present 3 different solution approaches for this problem. First, during the indexing process, equivalence classes of terms are obtained for normalization and the terms are indexed according to their equivalence classes. We also apply query expansion and document expansion methods during indexing and retrieval. According to the results of these tests, we highlight what is lost by writing Turkish with the standard Latin alphabet and how we can deal with this problem.

The rest of the paper is structured as follows. Section 2 gives brief information about the Turkish language and use of diacritics. Section 3 presents the solution approaches that we suggest for the problem at hand: normalization and equivalence classes, query expansion, and document expansion. Section 4 explains the details of the experiments conducted and the results that we gained. Finally, Section 5 summarizes the results and gives a look into future studies on this subject.

2. Preliminaries and definitions

The Turkish language belongs to the Altaic branch of the Ural-Altaic language family; there are many different Turkish dialectics, but this study focuses on the Turkish dialectic spoken in the Turkish Republic and Northern Cyprus. The Turkish alphabet consists of Latin letters, including 8 vowels and 21 constants. The constants are b, c, ç, d, f, g, ğ, h, j, k, l, m, n, p, r, s, ş, t, v, y, and z, and the vowels are a, e, ı, i, o, ö, u, and ü. Some of the Turkish letters are derived from the Latin alphabet with the addition of 4 diacritical marks. The diacritics used in the Turkish alphabet are the breve, umlaut/diaeresis, cedilla, dot/tittle, and circumflex.

- The breve mark is shaped like the bottom half of a circle and is placed over a letter. In the Turkish alphabet, the breve mark appears only with g/G and forms a new letter called "soft G." This letter, Ğ/ğ, lengthens the preceding vowel.
- The umlaut/diaeresis is a pair of dots placed over the letter that represents the affected vowel sound, namely O and Ö and U and Ü.
- The cedilla is a hook added under consonant letters c and s as a diacritical mark to modify their pronunciation. The letter ç represents the voiceless postalveolar affricate (as in English "church") in Turkish (as in "Çorum"). The letter ş represents the voiceless postalveolar fricative (as in "show") in Turkish (as in "şeker").
- The dot/tittle is a small distinguishing mark, such as a diacritic or the dot on an i or j. The Turkish alphabet includes 2 distinct versions of the letter I, with a dot and without a dot.
- The circumflex mark appears with 3 letters (â, î, and û) in Turkish; it softens the vowel. In contrast to the aforementioned diacritics, the letters with a circumflex are not considered as distinct letters.

These 5 diacritical marks compose the major points of Turkish language characteristics. Based on the given characteristics of the Turkish language, this study deals with the usage of diacritics in information retrieval. According to the given diacritical definitions, usage of diacritic letters without their umlauts, dots, cedilla, or breve turn them into new letters in the Latin alphabet. Table 1 shows those letters and their counterparts in the standard Latin alphabet.

Table 1. Turkish letters with diacritics and their counterparts in the standard Latin alphabet.

Turkish letters with diacritics	Standard Latin letters
ç/Ç	c/C
ğ/Ğ	g/G
ı/I	i/I
i/İ	i/I
ö/Ö	o/O
ş/Ş	s/S
ü/Ü	u/U

Characteristics of Turkish letters are similar to characteristics of the usage of diacritics in other languages. The effect of diacritics on the meaning of a word is not very clear in the English language; "cliché" and "cliche" or "naïve" and "naive" have the same meanings and have to be matched by the retrieval process. German has umlauts similar to Turkish umlauts, as in ö and ü. However, in German, umlauts (ä, ö, and ü) can be represented by a pair of letters (ae, oe, and ue), distinctively from Turkish. From the point of view of English and German, removing diacritics during the token normalization process seems quite reasonable [8]. However, in Polish, a word without a diacritic can become a totally different word and it loses its original meaning [4]. Similar to Polish, in Spanish, the meaning of a term changes with the usage of diacritics. For instance, "peña" means "a cliff" while "pena" means "sorrow." In Turkish, like Spanish and Polish, diacritics are a regular part

of the writing system and distinguish different sounds [9]. However, the main decision for processing diacritics is not based on the linguistic properties of a language. Diacritic usage depends on the habits of the user and computer system limitations. Thus, Manning et al. suggested normalizing tokens to remove diacritics. As for search engines, since 2006, Google determines for each query term whether there is a form with diacritics and searches for both of them, but the search can be forced to retrieve the exact form by adding a plus sign (+) to the term [10].

3. Token normalization for Turkish letters with diacritics

Token normalization is the process of canonicalizing tokens so that matches occur despite superficial differences in the character sequences of the tokens. The most standard way to normalize is to implicitly create equivalence classes, which are normally named after one member of the set [9]. Token normalization includes 3 specific structures: token, type, and term. A token is an instance of a sequence of characters that are grouped together as a useful semantic unit for processing. A type is the class of all tokens containing the same character sequence. A term is a type that is included in the system dictionary (normalized). During the process of breaking documents and queries up into tokens, some estimations have to be done to obtain the terms to be indexed. This is because, although the tokens in a collection have the same meaning, in general they do not have the exact same character sets. The reason for this is the occurrence of different forms of token or typing methods. In the retrieval process, these tokens that have different character sets but the same logical meaning have to be matched. For instance, if the term "USA" is searched for, documents that also contain "U.S.A." have to be matched. In this study, we simply replaced any Turkish letters with standard Latin letters, as given in Table 1. For example, the word "küçük" ("small" in English) in a query was converted to "kucuk" by equivalence classes. Therefore, the query processing will match any of the documents including the terms "küçük," "kücük," "kücuk," "küçuk," "kuçük," "kuçuk," "kucük," or "kucuk."

In this paper, we propose 3 different solution approaches to token normalization to improve the retrieval performance caused by the improper usage of diacritics: document expansion, query expansion, and equivalent classes. The combination of all of these methods helps us to signify the importance of the problem and how this performance loss can be regained when the mistyping of Turkish diacritics occurs in documents or queries. We first describe each solution briefly with respect to information retrieval. The usage of these methods for the problem solution is then given in detail.

Document expansion is a technique that aims to increase information retrieval effectiveness by adding extra terms during the document indexing process. Document expansion requires more space to store postings; at first sight, this seems to be a disadvantage, but by the development of modern technology, space costs have lost their importance and distinct posting lists have become preferable [9]. In this study, our document expansion approach simply adds the same terms with Latin letters. That is to say, if a document contains the term "büyük," or "large," this term will be indexed with "büyük" and "buyuk." As a result of this technique, any form of a term written with Turkish letters or without in a query can match the relevant documents.

Query expansion is based on the idea of adding extra terms to a query that stand for the meaning or the morphological structure of the query. Accordingly, the query expansion process may involve expanding the search query to match additional documents by finding synonyms of words or various morphological forms of words by stemming each word in the search query, or by fixing spelling errors or reweighting the terms in the original query. However, query expansion may also decrease the efficiency of retrieval time as the number of query terms increases. In this study, we performed the same expansion methods given for document expansion.

4. Experimentation

4.1. Test collection and generated test sets

The first goal of this study was to explicitly reveal the performance loss caused by improper usage of diacritics in both documents and queries for Turkish information retrieval. In order to do so, we needed a special document collection. We used a Turkish information retrieval test collection developed by the Bilkent Information Retrieval Group [11]. The test collection contains news articles and columns from 5 years (2001-2005) from the Turkish newspaper *Milliyet*, and its size is about 800 MB, containing 408,305 documents. We modified the original collection and obtained the following 3 different sets:

- **T**: The set in which the terms are intact and original with Turkish letters.
- **L**: All Turkish letters are replaced with standard Latin letters.
- **TL**: This set includes both the original form with Turkish letters plus equivalent forms with standard Latin letters.

In short, T, L, and TL stand for Turkish, Latin, and Turkish+Latin letters, respectively. T represents the original documents and queries without any modifications. L represents equivalence forms by token normalization of texts. TL denotes expansion and includes terms with both Turkish and Latin letters. We applied all of these modifications to both documents and queries. For clarification, we used a naming convention with *doc_query* pairs. For example, a test set named T_L indicates that the documents had Turkish letters and the queries had standard Latin letters. Applying 3 different modifications to both the documents and queries produced 9 different combinations. Table 2 shows each test set with example terms, their English meanings, and results of 2 different stemming options (NS for no stemming and F5 for truncating the terms after the first 5 characters).

Table 3 shows the distinct term counts in the lexicon. We see that the distinct term count of the L set is approximately 3% less than the term count of T for the NS option. This is because more than one term in the L set is represented by a single term after replacing the diacriticized Turkish letters with standard Latin letters. For example, "çiçek" ("flower" in English) and "çicek" (notice that the second c is without a cedilla) can both map to "cicek." This is a typical example of reduction in total term count. However, if a term is converted into an already existing term in the set, this causes a meaning loss due to the Latinization of the Turkish letters. For instance, the term "döndür" (meaning "rotate" in English) is represented by "dondur" ("freeze" in English) in the L set. Another possible result of this process is the representation 2 or more terms with a different term that is synthetic and has no meaning. A typical example of this issue is "açı" ("angle" in English) and "acı" ("pain" in English). They were both converted to a new synthetic term having no meaning, "aci." Thus, all of these changes result in a drop in the distinct term count.

The distinct term count in the L set is approximately 3% less than that of the original T set, as shown in Table 3. This means that the meaning loss is less than 3%. On the other hand, this difference is about 10% in the set with the F5 stemming option. These rates can be used to model the amount of information loss if we assume that the test collection does not contain misuse of diacritics. However, we observed that the test collection that we used did contain many errors. We thus conclude that the meaning loss is less than 3% and 10% due to the Latinization process with NS and F5 stemming options, respectively.

Table 2. Examples of terms from the collection, their English meaning, and the resulting forms in generated test sets with respect to different stemming options (NS and F5).

Collection terms	English meaning	Stemming approach	Index term in test sets		
			Turkish (T)	Turkish+Latin (TL)	Latin (L)
Anlaşmazlığım	my disagreement	NS	Anlaşmazlığım	anlaşmazlığım, anlasmazligim	anlasmazligim
		F5	Anlaş	anlaş, anlas	anlas
Kelebekler	butterflies	NS	Kelebekler	kelebekler	kelebekler
		F5	Keleb	keleb	keleb
YÖK	The Council of Higher Education	NS	YÖK	YÖK, YOK	YOK
		F5	YÖK	YÖK, YOK	YOK
Yok	to not exist	NS	Yok	yok	yok
		F5	Yok	yok	yok
Yaşamak	to live	NS	Yaşamak	yaşamak, yasamak	yasamak
		F5	Yaşam	yaşam, yasam	yasam
Yasamak	to legislate	NS	Yasamak	yasamak	yasamak
		F5	Yasam	yasam	yasam
Satranç	chess	NS	Satranç	satranç, satranc	satranc
		F5	Satra	satra	satra
İnternet	Internet	NS	İnternet	ınternet, internet	internet
		F5	İnter	ınter, inter	inter

Table 3. Distinct term counts in each test set.

Stemming option	Test sets		
	T	TL	L
NS	888,885	1,366,720	867,433
F5	166,595	187,293	147,671

4.2. Preprocessing and indexing

Files in the original test collection are based on UTF-8 encoding and include text written with proper Turkish letters. However, we observed the miswriting of many Turkish words. During the token normalization procedure in indexing and query processing, character casing is ignored, all letters are turned into their lowercase forms, and top word elimination is not applied. All apostrophes are accepted as term terminators, even if they are placed in the middle of a word. For example, the word "Ali'nin" is processed as 2 distinct words, "Ali" and "nin." The same methodology is also applied to numeric tokens; for example, "1.5" is indexed as "1" and "5." This approach leads to minimized lexicon length. If only the apostrophe in the middle of a term is accepted as a term terminator and all other marks that appear within a term, like in "1,073,210" or "7.editör," are left as part of the token, the lexicon grows in number of distinct term counts. We also compared our results for different retrieval models, namely the TF $\times$ IDF and BM25 document weighting models.

For the indexing and retrieval processes, we used the Terrier 2.1 platform, which was the first serious development in information retrieval in Europe [12]. It is a flexible, efficient, open-source search engine and it can deal with large-scale collections of documents. Terrier searches TREC and CLEF documents, is written in Java, and was developed at the Department of Computing Science of the University of Glasgow [13].

4.3. Experimentation naming convention

We ran a series of experiments in order to validate our research hypothesis and to evaluate how the proposed solution approach works for this issue. Each experiment was named with a notation that was a combination of the 5 basic parameters: the stemming approach, document alphabet, query alphabet, weighting, and query length. Table 4 depicts all of the parameters of the experimentation, their order, their abbreviations, and short descriptions. For instance, a test name of *F5_T_TL_BM_M* shows that the experiment was generated by the F5 stemming algorithm, the documents were in their original form (T), queries were expanded containing both Turkish and Latinized version, the BM25 weighting model was used, and the query included only medium-length queries. As another example, *NS_L_TL_TF_ALL* means that no stemming method was applied during the token normalization process, documents were written with standard Latin letters, the query text contained both Turkish and Latin letters, the TF × IDF weighting approach was used, and queries were in the longest forms, consisting of titles, descriptions, and narratives.

Table 4. The experiment's naming convention, the abbreviations used in notation, and explanations.

Order	Phase/step	Parameter	
		Abbreviation	Description
1	Stemming	NS	No stemming
		F5	First 5 letters of each token
2	Documentindexing	T	Documents with **T**urkish letters: original documents
		TL	Documents with both **T**urkish and standard **L**atin letters: document expansion
		L	Documents with standard **L**atin letters: equivalent classes
3	Query	T	Queries with **T**urkish letters: original query
		TL	Queries with both **T**urkish and standard **L**atin letters: query expansion
		L	Queries with standard **L**atin letters: equivalent classes
4	Weighting	TF	TF×IDF
		BM	BM25
5	Query length	S	Titles as short queries
		M	Descriptions as medium queries
		L	Narratives as long queries
		SM	Titles + descriptions
		ALL	Titles + descriptions + narratives

4.4. Baseline retrieval

In the comparison of the evaluation results of all test parameters, the test results for NS and F5 for middle-length queries (Qm) by TF × IDF were chosen as baseline results for our test collection. Table 5 shows our baseline retrieval results versus the original results from referenced paper [11] comparatively in the *bpref* measure with NS and F5 options.

Table 5. Comparison of baseline retrieval results in *bpref* values for Qm by TF × IDF weighting versus the original results from the referenced paper.

NS		F5	
Bilkent (MF8)	Our study	Bilkent (MF8)	Our study
0.3255	0.3579	0.4322	0.4471

The details of the methods of the test results taken from the referenced paper can be explained by matching function approximation. In the referenced paper, different matching functions were applied during the retrieval process. These functions were generated by the combination of the 3-term weighting component: the term frequency component (TFC), collection frequency component (CFC), and normalization component (NC), both for documents and queries. The combination of these factors depends on the idea of multiplying the respective weights of these factors. Multiplying these factors for documents and queries provides the evaluation of term weight in a document (w_{dk}) or a query (w_{qk}).

The similarity of a document and a query is formalized as follows [14]:

$$Sim(Doc,\ Q) = \sum_{k=1}^{m} w_{dk} * w_{qk}. \tag{1}$$

All of these weighting components have different possibilities; the combination of these possibilities makes the difference in matching functions. Another matching strategy was applied in the referenced paper, called MF8 or matching function 8. The results given in Table 6 were obtained with that matching function [15]. MF8 calculates matching values for document d_j for search query Q, as follows:

$$MF8 = \sum_{i \in Q} (\frac{1 + \ln f_{dt}}{\sqrt{D}} * (f_{qt} * \ln(1 + \frac{N}{f_t})), \tag{2}$$

where f_{dt} is the frequency of term t in document d_j, D is the total number of term occurrences in d_j, and these parameters form the document factor of the formula. In addition to the document factor of the formula, the query factor consists of parameters f_{qt} (the frequency of term t in the query Q), N (the total number of documents), and f_t (the frequency of term t in the entire collection).

In their original work, Can et al. indicated that MF8 is particularly suitable for dynamic environments because of the effect of query term weights in the second component [11]. They reported that MF8 gave the best results for NS and F5. Hence, we accepted the MF8 function results as a base retrieval for comparison of our evaluation results.

4.5. Experimentation results

The starting point of this study was to determine the loss of performance by using standard Latin alphabet letters instead of diacritized Turkish letters in document and/or queries. Table 6 shows the exact evaluation results for the tests, highlighting the performance loss in the mistyping of Turkish letters.

As shown in Table 6, the best value for the NS approach is the result of documents and queries with proper Turkish diacritics. If we take the value of the test *NS_T_T_M_TF* (no stemming method applied, documents and queries in their original form), we get the *bpref* evaluation result of 0.3579 as the basis. The ratio between the initial point and the test results of *NS_L_T_M_TF*, which simulates the nonexistence of Turkish letters in documents, or the test results of *NS_T_L_M_TF*, which represents the nonexistence of Turkish letters in queries, gives us the exact information that this study aims for. While the retrieval efficiency of the *bpref* value is about 0.3579 for the initial point, it decreases to 0.2462 if the documents do not contain any Turkish letters but the queries are typed using proper Turkish letters. A similar situation also occurred with a value of 0.2466 for the nonexistence of Turkish letters in queries but not in documents.

Table 6. Exact evaluation results in *bpref* values for the test sets for medium-length queries and TF × IDF weighting.

		Document					
		T		TL		L	
		NS	F5	NS	F5	NS	F5
Query	T	0.3579	0.4471	0.3571	0.4445	0.2462	0.3269
	TL	0.3528	0.4247	0.3255	0.4179	0.3505	0.4226
	L	0.2466	0.3254	0.3542	0.4414	0.3552	0.4435

As shown in Tables 7 and 8, our research hypothesis, which is that improper usage of Turkish diacritics either in documents or queries causes a large drop in performance, is supported with rates of 32.2% and 31.09% for the NS approach and 26.88% and 27.21% for the F5 stemming option. We conducted a set of t-tests and found that all differences were statistically significant ($P < 0.000002$). These observations confirm our previously stated research statement and the major motivation of this study.

Table 7. The loss of performance in *bpref* values of retrieval results according to the initial point for NS using query form Qm by the TF × IDF weighting model.

		Document		
		T	TL	L
Query	T	-	0.22%	32.2%
	TL	1.42%	9.05%	2.06%
	L	31.09%	1.03%	0.75%

Table 8. The loss of performance in *bpref* values of retrieval results according to the baseline retrieval for F5 using query form Qm by the TF × IDF weighting model.

		Document		
		T	TL	L
Query	T	-	0.58%	26.88%
	TL	5.01%	6.53%	5.47%
	L	27.21%	1.27%	0.80%

In order to see how the change in retrieval performance is distributed among individual queries, we further investigated the results to see whether the misuse of diacritics evenly affected the retrieval performance of each query or whether the loss in retrieval performance was only due to the change in a few numbers of queries. We observed a performance loss in 54 and 56 queries out of 72 for the NS and F5 options, respectively. These findings show that the performance loss was evenly distributed throughout the queries.

As Table 6 shows the *bpref* values of NS and F5 for the TF × IDF weighting model, Table 9 shows the *bpref* values when the NS and F5 stemming options and the BM25 weighting model are used. The rates given in Tables 10 and 11 were obtained from the results in Table 9.

Table 9. *bpref* values of NS and F5 for Qm by the BM25 model.

		Document					
		T		TL		L	
	Stemming→	NS	F5	NS	F5	NS	F5
Query	T	0.3589	0.4559	0.3575	0.4544	0.248	0.3325
	TL	0.3555	0.4345	0.3263	0.4274	0.3528	0.4336
	L	0.247	0.3299	0.3548	0.4514	0.3562	0.4536

Tables 10 and 11 show, similarly to Tables 7 and Table 8, that improper use of Turkish letters in documents and/or queries causes performance loss in information retrieval with respect to the stemming options for the BM25 weighting model.

Table 10. The loss of performance in *bpref* values of retrieval results according to the initial point for NS using query form Qm by the BM25 weighting model.

		Document		
		T	TL	L
Query	T	-	0.39%	30.89%
	TL	0.94%	9.08%	1.69%
	L	31.17%	1.42%	0.75%

Table 11. The loss of performance in *bpref* values of retrieval results according to the initial point for F5 using query form Qm by the BM25 weighting model.

		Document		
		T	TL	L
Query	T	-	0.32%	27.06%
	TL	4.69%	6.25%	4.89%
	L	27.63%	0.98%	0.50%

Tables 7 and 11 show that the BM25 weighting model produces worse results for document expansion (TL) and token normalization by equivalent classes (L) for queries in the BM25 model compared to the TF$\times$IDF model for the NS approach. Tables 7, 8, 10, and 11 prove that although the rates change with stemming methods and other weighting models, there is a clear-cut performance loss in the retrieval performance caused by improper usage of diacritized Turkish letters.

On the other hand, the query expansion technique gave better results for the T and L test sets. Applying query expansion and document expansion at the same time produced a larger loss rate, as the rates were 1.42% and 2.06% for test sets T and L but 9.05% for test set TL for the NS approach in the TF$\times$IDF weighting model. Similar to NS in TF$\times$IDF, F5 in TF$\times$IDF degraded performance by rates of 5.01% and 5.47% for the T and L test sets, but 6.53% for TL. Differences in the negative impact of query expansion for different test sets were smaller for F5 by 5.01% for T, 5.47% for L, and 6.53% for TL. The difference in the rates for different test sets for NS were more appreciable, by 1.42%, 2.06%, and 9.05% for datasets T, L, and TL, respectively.

As seen in Tables 6-11, document expansion led to a smaller loss in performance and gave more stable results for each lexicon regardless of the stemming method and weighting model. The most important point of document expansion is not to apply document and query expansion at the same time. However, document expansion gave better results for test sets T and L at all times.

In addition to document and query expansion techniques, another approximation to the solution of the mistyping of Turkish letters is applying token normalization to obtain synthetic types for both queries and documents. This technique gives better results than all other techniques applied in this paper, except if the documents are expanded (TL) and queries are typed in Turkish letters (T). This assumption is valid for all stemming approaches and weighting methods applied in this study. The difference between the results of *TL_T* and *L_L* were not very noticeable. The differences of the ratios of these techniques were 0.53%, 0.22%, 0.36%, and 0.18% for *NS_TF*, *F5_TF*, *NS_BM25*, and *F5_BM*, respectively.

Table 12 shows the number of retrieved documents when either the documents or queries do not include any Turkish letters. Although all other query techniques retrieve at least 1000 documents for all queries, the *T_L* and *L_T* tests retrieved fewer than 1000 documents for some queries for NS. Most of the terms in the queries did not match because of their new synthetic forms.

Table 12. Retrieval performance in different evaluation measures for NS, Qm, and the given test parameters.

Doc	Query	Weight	#ret	#rel_ret	Map	gm_ap	R-prec	bpref	recip_rank
T	T	BM	72,000	4760	0.2793	0.167	0.3195	0.3589	0.7164
T	T	TF	72,000	4747	0.2763	0.1644	0.319	0.3579	0.7152
T	TL	BM	72,000	4731	0.2694	0.1548	0.3134	0.3555	0.6435
T	TL	TF	72,000	4686	0.2647	0.1504	0.3085	0.3528	0.6381
T	L	BM	70,292	3045	0.16	0.0314	0.1949	0.247	0.4963
T	L	TF	71,208	3014	0.1573	0.0316	0.1942	0.2466	0.4906
TL	T	BM	72,000	4762	0.2785	0.1663	0.3207	0.3575	0.7311
TL	T	TF	72,000	4739	0.2756	0.1633	0.3179	0.3571	0.7268
TL	TL	BM	72,000	4393	0.2335	0.1269	0.2788	0.3263	0.6851
TL	TL	TF	72,000	4385	0.231	0.1256	0.2768	0.3255	0.6874
TL	L	BM	72,000	4696	0.2744	0.1649	0.3192	0.3548	0.7263
TL	L	TF	72,000	4685	0.2719	0.1623	0.3166	0.3542	0.724
L	T	BM	70,137	3060	0.1662	0.0339	0.1997	0.248	0.5384
L	T	TF	71,118	3024	0.1637	0.0337	0.1986	0.2462	0.5362
L	TL	BM	72,000	4671	0.27	0.1609	0.3151	0.3528	0.6918
L	TL	TF	72,000	4641	0.2653	0.1564	0.3111	0.3505	0.6783
L	L	BM	72,000	4692	0.2745	0.1652	0.3181	0.3562	0.7215
L	L	TF	72,000	4682	0.2717	0.1626	0.316	0.3552	0.7143

Table 13 shows that although the retrieval efficiencies were still worse for tests *T_L* and *L_T*, they did retrieve 1000 documents for each query, and as the number of relevant documents increased, efficiency improved with the F5 stemming method. This improvement can be seen for each different evaluation measure for both the TF $\times$ IDF and BM25 weighting models.

Performance changes for precision at different recall values are given in Figures 1 and 2 for stemming options NS and F5 in TF $\times$ IDF for medium-length queries. It is clear that the major difference in the precision-recall graph occurs when diacritics are missing in either documents or in queries.

4.6. Performance for different stemming methods

Tables 14 and 15 prove that there is always an increase in efficiency when the F5 stemming method is applied. The increase rates are better for the BM25 weighting model, in the range of 2% and 3%. The increase rates are also better for the tests that have worse efficiency with NS and F5. For instance, the F5 method has a better impact on tests *T_L* and *L_T* with improvement rates of 33.56% and 34.07%; those tests have lower performance because of the mistyping of Turkish letters, as compared to other tests like *T_T*. According to the given test results, F5 always gave better results than NS under different indexing, retrieval, and weighting techniques, although the retrieval efficiency was affected negatively by these parameters.

Table 13. Retrieval performance in different evaluation measures for F5, Qm, and the given test parameters.

Doc	Query	WM	#ret	#rel_ret	map	gm_ap	R-prec	bpref	recip_rank
T	T	BM	72,000	5793	0.3433	0.2497	0.3734	0.4559	0.7928
T	T	TF	72,000	5662	0.3288	0.2387	0.3614	0.4471	0.7862
T	TL	BM	72,000	5460	0.3065	0.211	0.3517	0.4345	0.7455
T	TL	TF	72,000	5324	0.2938	0.1988	0.3403	0.4247	0.7158
T	L	BM	72,000	4136	0.1991	0.0565	0.2379	0.3299	0.503
T	L	TF	72,000	4033	0.1927	0.0571	0.2336	0.3254	0.495
TL	T	BM	72,000	5711	0.3374	0.2428	0.3699	0.4544	0.7743
TL	T	TF	72,000	5594	0.3235	0.2334	0.3586	0.4445	0.7968
TL	TL	BM	72,000	5299	0.3024	0.1956	0.3449	0.4274	0.7709
TL	TL	TF	72,000	5173	0.2888	0.1853	0.3305	0.4179	0.7706
TL	L	BM	72,000	5670	0.3342	0.2403	0.3659	0.4514	0.7883
TL	L	TF	72,000	5558	0.32	0.2308	0.3554	0.4414	0.7898
L	T	BM	72,000	4199	0.2207	0.062	0.254	0.3325	0.5597
L	T	TF	72,000	4082	0.2122	0.0611	0.2483	0.3269	0.5668
L	TL	BM	72,000	5455	0.3144	0.2175	0.355	0.4336	0.8034
L	TL	TF	72,000	5316	0.2983	0.2022	0.3418	0.4226	0.7645
L	L	BM	72,000	5744	0.3398	0.2463	0.3684	0.4536	0.8099
L	L	TF	72,000	5628	0.3239	0.2352	0.3592	0.4435	0.7937

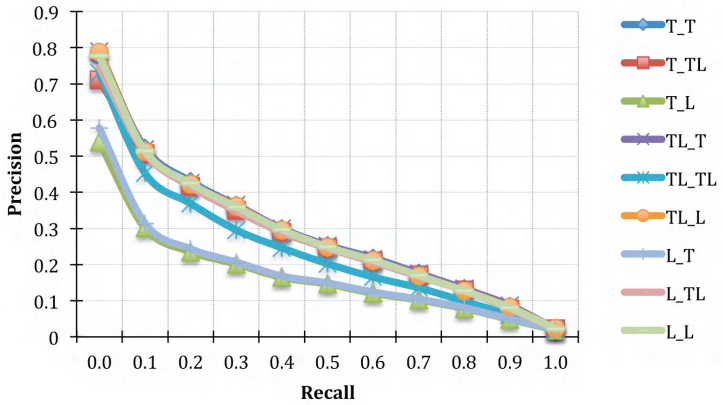

Figure 1. Precision-recall graph of test results of NS in TF × IDF for Qm.

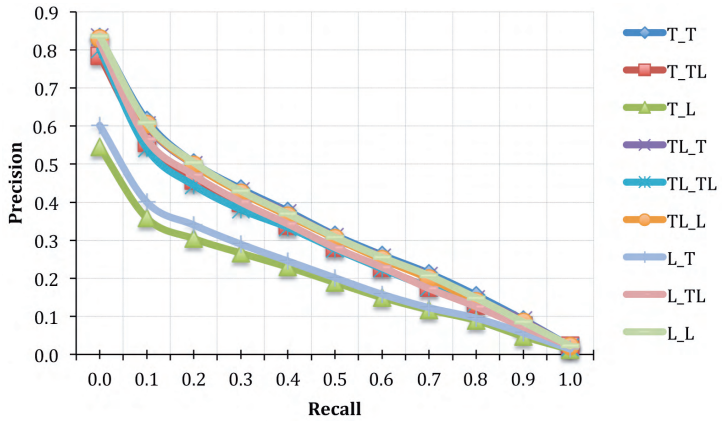

Figure 2. Precision-recall graph of test results of F5 in TF × IDF for Qm.

Table 14. The rate of the retrieval performance in *bpref* with F5 to NS stemming options using medium-length queries and TF × IDF weighting. Values were derived from the values of Table 6.

		Document		
		T	TL	L
		F5/NS	F5/NS	F5/NS
Query	T	24.92%	24.47%	32.77%
	TL	20.37%	28.38%	20.57%
	L	31.95%	24.61%	24.85%

Table 15. The rate of the retrieval performance in *bpref* with F5 to NS stemming options using medium-length queries and BM25 weighting. Values were derived from the values of Table 9.

		Document		
		T	TL	L
		F5/NS	F5/NS	F5/NS
Query	T	27.02%	27.10%	34.07%
	TL	22.22%	30.98%	22.90%
	L	33.56%	27.22%	27.34%

4.7. Performance results of different weighting models

As seen from Tables 16 and 17, there was not a significant change in *bpref* values for different weighting models. The results of the tests generated for TF × IDF and BM25 are close to each other because of the Robertson term frequency (TF) factor in the TF × IDF implementation. Implementation of the Robertson TF factor generates a different formula than the standard TF × IDF definition.

Table 16. *bpref* values of NS for the query form Qm for weighting models TF × IDF and BM25.

		Document					
		T		TL		L	
		TF	BM25	TF	BM25	TF	BM25
Query	T	0.3579	0.3589	0.3571	0.3575	0.2462	0.248
	TL	0.3528	0.3555	0.3255	0.3263	0.3505	0.3528
	L	0.2466	0.247	0.3542	0.3548	0.3552	0.3562

Table 17. *bpref* values of F5 for the query form Qm for weighting models TF × IDF and BM25.

		Document					
		T		TL		L	
		TF	BM25	TF	BM25	TF	BM25
Query	T	0.4471	0.4559	0.4445	0.4544	0.3269	0.3325
	TL	0.4247	0.4345	0.4179	0.4274	0.4226	0.4336
	L	0.3254	0.3299	0.4414	0.4514	0.4435	0.4536

Table 18. Information performance results in *bpref* for di erent stemming options, test sets, and query lengths for weighting models TF× IDF and BM25.

			Document											
			NS/TF-IDF			F5/TF-IDF			NS/BM25			F5/BM25		
		Query length	T	TL	L	T	TL	L	T	TL	L	T	TL	L
Query	T	S	0.3877	0.3841	0.2616	0.4637	0.4600	0.3323	0.3872	0.3838	0.2618	0.4633	0.4597	0.3311
		M	0.3579	0.3571	0.2462	0.4471	0.4445	0.3269	0.3589	0.3575	0.248	0.4559	0.4544	0.3325
		L	0.3665	0.3657	0.2765	0.4308	0.4270	0.3530	0.3683	0.3671	0.2773	0.4411	0.4353	0.3562
		ALL	0.4562	0.4531	0.3205	0.5259	0.5222	0.3985	0.4569	0.4544	0.3202	0.5282	0.5256	0.3985
		SM	0.4366	0.434	0.2938	0.5242	0.5204	0.3800	0.436	0.4336	0.2934	0.5256	0.5227	0.3824
	TL	S	0.3859	0.358	0.3781	0.4407	0.4429	0.4377	0.3859	0.3568	0.3784	0.4409	0.4442	0.4389
		M	0.3528	0.3255	0.3505	0.4247	0.4179	0.4226	0.3555	0.3263	0.3528	0.4345	0.4274	0.4336
		L	0.3636	0.3407	0.3633	0.4086	0.3947	0.4106	0.3657	0.3412	0.3657	0.4178	0.4052	0.4192
		ALL	0.4534	0.4389	0.4548	0.5060	0.5029	0.5051	0.4552	0.4379	0.4564	0.5109	0.5076	0.5102
		SM	0.4334	0.4056	0.4344	0.5000	0.4947	0.4983	0.4338	0.4041	0.4342	0.5039	0.4975	0.5036
	L	S	0.2466	0.3767	0.3794	0.3208	0.4573	0.4591	0.2462	0.3766	0.3792	0.3192	0.4572	0.4588
		M	0.2466	0.3542	0.3552	0.3254	0.4414	0.4435	0.247	0.3548	0.3562	0.3299	0.4514	0.4536
		L	0.2724	0.3651	0.365	0.3451	0.4251	0.4291	0.2741	0.366	0.3674	0.3492	0.4338	0.4397
		ALL	0.3167	0.4538	0.4567	0.3934	0.5199	0.5229	0.3166	0.4555	0.4578	0.3932	0.5235	0.5259
		SM	0.2936	0.4339	0.4369	0.3784	0.5166	0.5201	0.2927	0.4335	0.4359	0.3796	0.5193	0.5221

4.8. IR performance results with respect to different query lengths

The test results illustrate the effects of query length on retrieval results with the combination of other defined parameters like stemming and weighting models. These results depend on the new synthetic forms of a query. An example query in both original and expanded from for different query lengths is given in the Appendix.

According to test results for NS and the TF × IDF weighting model in Table 18, all test results show that the most efficient test was run for the combination of all query lengths, called *ALL*. This appreciable diversification for query length *ALL* occurs because of the increase of key frequency in queries of terms that have low term frequency in the collection. The behavior of query lengths S, M, and L differ for different document indexing and query processing methods. Short queries give the best results for most cases except test *T_L*. Table 18 also shows that although BM25 gives better results for all test sets, there is not a significant change when the weighting model changes. It is clear that F5 always gives better results than NS regardless of query lengths.

5. Conclusions

The aim of this study was to explicitly highlight the performance loss in information retrieval caused by misuse of diacriticized Turkish letters intermixed with standard Latin letters. In other words, this paper examined the results of improper usage of Turkish diacritics in documents and/or queries with respect to information retrieval and provided solutions for this degradation. First, we removed the diacritics in the documents and/or the queries to determine the performance loss, and then the retrieval process was carried out to observe the difference and provide a comparison with baseline retrieval results. Document expansion, query expansion, and equivalence class techniques were then applied to obtain the initial information retrieval performance. In addition to the document indexing and query processing techniques, different stemming methods, weighting models, and query lengths were compared with the baseline retrieval results.

Several major finding of this study can be summarized as follows:

- The retrieval performance significantly drops if the documents and queries are written with different alphabets. We obtained a 32.05% loss in retrieval performance using NS and the TF × IDF weighting model when Turkish letters with diacritics were not used in either documents or queries. We also conducted t-tests and showed that the differences were statistically significant. This was expected, and it was the motivation of our study.

- Document expansion, which is the indexing of original Turkish text together with expanded Latin equivalence, produces good results for the research problem defined. On the other hand, applying query expansion alone, without document expansion, is also useful, but it performs slightly worse than the former approach.

- Expanding both queries and documents at the same time does not work; the results were poor. This may be the result of the blind expansion process, which is shown in the literature to causes severe drifting [16].

- The final finding is quite interesting and indicates that simply using standard Latin letters with both the queries and documents is almost as good as using the original documents and queries. This may be related to the information loss caused by replacing Turkish letters with standard Latin letters. We have shown that this loss is less than 3% and 10% for the NS and F5 stemming approaches, respectively. These rates

are valid only if we assume that test collection does not contain any misuse of diacritics. However, we have already encountered many misuses of diacritics in the test collection.

Our work can be extended in several ways in the future. First, the information loss as a result of the Latinization process may also cause a meaning loss. However, the exact amount of the meaning loss should be examined on a dataset specifically prepared with queries and collections for that purpose. Second, the improper use of diacritics is also considered as a subproblem of spelling correction in Turkish [17]. We see that both problems are akin to each other and we suggest that it would be a good research topic for a new study to see how automatic spelling correction helps with this issue. Lastly, misuse of Turkish letters with diacritics is also important for other subdisciplines such as natural language processing and information extraction from Turkish text [18]. Henceforth, it would be good to apply the findings of this study to these fields and see the results.

References

[1] J. Bar-Ilan, T. Gutman, "How do search engines handle non-English queries?", Proceedings of the Twelfth International World Wide Web Conference, Budapest, 2003.

[2] G. Grefenstette, J. Nioche, "Estimation of English and non-English language use on the WWW", Proceedings of Recherche d'Information Assistée par Ordinateur, pp. 237-246, 2000.

[3] M. Soroka, "Web search engines for Polish information retrieval: questions of search capabilities and retrieval performance", The International Information & Library Review, Vol. 32, pp. 87-98, 2000.

[4] K. Choros, "Testing the effectiveness of retrieval to queries using Polish words with diacritics", Lecture Notes in Computer Science, Vol. 3528, pp. 101-106, 2005.

[5] H. Moukdad, "Lost in cyberspace: how do search engines handle Arabic queries?", Proceedings of the 32nd Annual Conference of the Canadian Association for Information Science, 2004.

[6] A.M. Daoud, "Morphological analysis and diacritical Arabic text compression", International Journal of ACM, Vol. 1, pp. 41-47, 2010.

[7] T. Bitirim, Y. Tonta, H. Sever, "Information retrieval effectiveness of Turkish search engines", Lecture Notes in Computer Science, Vol. 2457, pp. 93-103, 2002.

[8] M. Braschler, B. Ripplinger, P. Schauble, "Experiments with the Eurospider retrieval system for CLEF 2001", Lecture Notes in Computer Science, Vol. 2406, pp. 102-110, 2002.

[9] C.D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge, Cambridge University Press, 2008.

[10] Google Webmaster, "How search results may differ based on accented characters and interface languages", 2006. Retrieved 10 May 2010 from: http://googlewebmastercentral.blogspot.com/2006/08/how-search-results-may-differ-based-on.html.

[11] F. Can, S. Kocberber, E. Balcik, C. Kaynak, H.C. Ocalan, O.M. Vursavas, "Information retrieval on Turkish texts", Journal of the American Society for Information Science and Technology, Vol. 59, pp. 407-421, 2008.

[12] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, C. Lioma, "Terrier: a high performance and scalable information retrieval platform", Proceedings of the 2nd International Workshop on Open Source Information Retrieval, 2006.

[13] University of Glasgow, Terrier, 2010. Retrieved 8 February 2010 from http://www.terrier.org.

[14] G. Salton, C. Buckley, "Term weighting approaches in automatic text retrieval", Information Processing and Management, Vol. 24, pp. 513-523, 1988.

[15] X. Long, T. Suel, "Optimized query execution in large search engines with global page ordering", Proceedings of the 29th Very Large Data Bases Conference, 2003.

[16] M. Mitra, A. Singhal, C. Buckley, "Improving automatic query expansion", Proceedings of SIGIR, pp. 206-214, 1998.

[17] K. Oflazer , C. Güzey, "Spelling correction in agglutinative languages", Proceedings of the Fourth Conference on Applied Natural Language Processing, 1994.

[18] İ. Pehlivan, Z. Orhan, "Automatic knowledge extraction for filling in biography forms from Turkish texts", Turkish Journal of Electrical Engineering and Computer Sciences, Vol. 19, pp. 59-71, 2011.

Appendix

Table A1. A standard query #1 for different lengths (short, medium, long).

Query length	Query text
S	Kuş Gribi
M	Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar.
L	Kuş gribi ile alakalı her türlü bilginin elde edilebileceği bir doküman olmalı. Hastalığının tanımını, bulaşma yollarını, belirtilerini, varsa tedavi yollarını ve buna benzer birçok konuyu okuyucuya açıklayan nitelikte bir doküman.

Table A2. Expanded form of query in Table A1 for varying lengths.

Alphabet	Query length	Query
T	S	Kuş Gribi
	M	Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar.
	L	Kuş gribi ile alakalı her türlü bilginin elde edilebileceği bir doküman olmalı. Hastalığının tanımını, bulaşma yollarını, belirtilerini, varsa tedavi yollarını ve buna benzer birçok konuyu okuyucuya açıklayan nitelikte bir doküman.
	ALL	Kuş Gribi Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar. Kuş gribi ile alakalı her türlü bilginin elde edilebileceği bir doküman olmalı. Hastalığının tanımını, bulaşma yollarını, belirtilerini, varsa tedavi yollarını ve buna benzer birçok konuyu okuyucuya açıklayan nitelikte bir doküman.
	SM	Kuş Gribi Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar.
TL	S	Kuş Gribi Kus
	M	Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar. Kus nasil bulasir sorularina dokumanlar
	L	Kuş Gribi Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar. Kuş gribi ile alakalı her türlü bilginin elde edilebileceği bir doküman olmalı. Hastalığının tanımını, bulaşma yollarını, belirtilerini, varsa tedavi yollarını ve buna benzer birçok konuyu okuyucuya açıklayan nitelikte bir doküman. kus nasil bulasir sorularina dokumanlar alakali turlu edilebilecegi dokuman olmali hastaliginin tanimini bulasma yollarini yollarini bircok aciklayan dokuman
	ALL	Kuş Gribi Kus Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar. Kus nasil bulasir sorularina dokumanlar Kuş Gribi Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar. Kuş gribi ile alakalı her türlü bilginin elde edilebileceği bir doküman olmalı. Hastalığının tanımını, bulaşma yollarını, belirtilerini, varsa tedavi yollarını ve buna benzer birçok konuyu okuyucuya açıklayan nitelikte bir doküman. kus nasil bulasir sorularina dokumanlar alakali turlu edilebilecegi dokuman olmali hastaliginin tanimini bulasma yollarini yollarini bircok aciklayan dokuman
	SM	Kuş Gribi Kus Kuş gribi nedir, nasıl bulaşır, belirtileri nelerdir sorularına cevap olabilecek dokümanlar. Kus nasil bulasir sorularina dokumanlar
L	S	Kus Gribi
	M	Kus gribi nedir, nasil bulasir, belirtileri nelerdir sorularina cevap olabilecek dokumanlar.
	L	Kus gribi ile alakali her turlu bilginin elde edilebilecegi bir dokuman olmali. Hastaliginin tanimini, bulasma yollarini, belirtilerini, varsa tedavi yollarini ve buna benzer bircok konuyu okuyucuya aciklayan nitelikte bir dokuman.
	ALL	Kus Gribi Kus gribi nedir, nasil bulasir, belirtileri nelerdir sorularina cevap olabilecek dokumanlar. Kus gribi ile alakali her turlu bilginin elde edilebilecegi bir dokuman olmali. Hastaliginin tanimini, bulasma yollarini, belirtilerini, varsa tedavi yollarini ve buna benzer bircok konuyu okuyucuya aciklayan nitelikte bir dokuman.
	SM	Kus Gribi Kus gribi nedir, nasil bulasir, belirtileri nelerdir sorularina cevap olabilecek dokumanlar.