GRAS: An Effective and Efficient Stemming Algorithm for Information Retrieval

Article *in* ACM Transactions on Information Systems · December 2011

DOI: 10.1145/2037661.2037664 · Source: DBLP

Jiaul H. Paik
Indian Statistical Institute
7 PUBLICATIONS **101** CITATIONS

Mandar Mitra
Indian Statistical Institute
83 PUBLICATIONS **6,567** CITATIONS

Swapan Kumar Parui
Indian Statistical Institute
159 PUBLICATIONS **2,266** CITATIONS

GRAS: An Effective and Efficient Stemming Algorithm for Information Retrieval

JIAUL H. PAIK, MANDAR MITRA, and SWAPAN K. PARUI, Indian Statistical Institute
KALERVO JÄRVELIN, University of Tampere

A novel graph-based language-independent stemming algorithm suitable for information retrieval is proposed in this article. The main features of the algorithm are retrieval effectiveness, generality, and computational efficiency. We test our approach on seven languages (using collections from the TREC, CLEF, and FIRE evaluation platforms) of varying morphological complexity. Significant performance improvement over plain word-based retrieval, three other language-independent morphological normalizers, as well as rule-based stemmers is demonstrated.

Categories and Subject Descriptors: H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods; linguistic processing*

General Terms: Algorithms, Experimentation, Performance

Additional Key Words and Phrases: Corpus, stemming, suffix

ACM Reference Format:
Paik, J. H., Mitra, M., Parui, S. K., and Järvelin, K. 2011. GRAS: An effective and efficient stemming algorithm for information retrieval. ACM Trans. Inf. Syst. 29, 4, Article 19 (November 2011), 24 pages.
DOI = 10.1145/2037661.2037664 http://doi.acm.org/10.1145/2037661.2037664

1. INTRODUCTION

The main goal of this article is to describe a novel graph-based language-independent stemming algorithm (GRAS for GRAph-based Stemmer) for information retrieval. The following features make our algorithm attractive: (1) retrieval effectiveness, (2) generality, that is, its language-independent nature, and (3) low computational cost.

The usefulness of stemming in ad hoc retrieval has been studied over the years for several languages. Although experiments with English data show mixed results [Harman 1991; Hull 1996; Krovetz 1993], retrieval performance for morphologically more complex languages has benefited consistently and significantly when morphological normalization is done. For languages such as Hungarian and Czech, 30–40% improvement in performance was obtained through stemming (as compared to plain word-based retrieval) [Savoy 2008; Majumder et al. 2008]. Generally stemming algorithms are considered as recall-enhancing devices. Nevertheless, for morphologically complex languages, their ability to improve precision at the top of the ranked list is another notable feature which is important in text retrieval [Xu and Croft 1998].

Stemming algorithms can be broadly classified into two categories, namely rule-based [Porter 1997] and statistical [Oard et al. 2001; Bacchin et al. 2005; Majumder et al. 2007]. Rule-based stemmers encode a set of language-specific rules, whereas statistical

Author's addresses: J. Paik, M. Mitra, and S. Parui, Indian Statistical Institute, 203 B. T. Road, Kolkata-108, India; email: jia.paik@gmail.com; K. Järvelin, University of Tampere, Tampere, Finland.

DOI 10.1145/2037661.2037664 http://doi.acm.org/10.1145/2037661.2037664

stemming employs statistical information from a large corpus of a given language to learn the morphology. An immediate advantage of statistical stemmers over the rule-based ones is that they obviate language-specific expertize, without significantly sacrificing retrieval accuracy.

Most available statistical-stemming algorithms take a long time to construct a stemmer from a given corpus [Majumder et al. 2007; Goldsmith 2001; Bacchin et al. 2005]. Therefore, our main aim is to develop a computationally simple stemming algorithm which performs effectively in the retrieval task.

We focus primarily on suffixing languages, that is, languages in which inflected words are usually formed from the root by the process of suffixation. For such languages, morphologically-related words typically share a long common prefix. Since the main goal of a stemming algorithm is to identify such morphologically related words and then map them to a canonical representative, we start by considering word pairs of the form $\langle w_1 = ps_1, w_2 = ps_2 \rangle$ that share a sufficiently long common prefix p. It is not unlikely that the pair of suffixes s_1 and s_2 that remain after the common prefix is removed are linguistically valid suffixes, but we look for additional evidence that s_1 and s_2 are indeed a pair of linguistically valid suffixes. This is done by checking if other word pairs also have a common initial part followed by these suffixes. Such pairs would be of the form $\langle w'_1 = p's_1, w'_2 = p's_2 \rangle$. We regard s_1 and s_2 as a pair of candidate suffixes only if we find a sufficiently large number of word pairs of this form. The key idea here is that suffixes are considered in pairs, rather than individually.

Once candidate suffix pairs are identified, we look for pairs of words that are potentially morphologically related. Two words are regarded as possibly related if (1) they share a non-empty common prefix, and (2) the suffix pair that remains after the removal of the common prefix is a candidate pair identified in the first phase of our approach. These word relationships can be modeled by a graph, with the words being mapped to nodes, and potentially related word pairs being connected by edges. We next identify *pivot* nodes—words that are connected by edges to a large number of other words. In the final step, a word that is connected to a pivot is put in the same class as the pivot if it shares many common neighbours with the pivot, that is, if the words that it is related to are also related to the pivot. Once such word classes are formed, stemming is done by mapping all the words in a class to the pivot for that class.

We experimented with five European (Hungarian, Czech, English, French, and Bulgarian) and two Indian (Marathi and Bengali) languages of very different language families (Romance, East European, Eastern and Western Indo Aryan) and varying morphological complexity (simpler languages like English and French to more complex agglutinative languages like Hungarian). Using test collections provided by the TREC, CLEF, and FIRE evaluation forums, we found that the proposed algorithm outperformed rule-based stemmers, three statistical methods (YASS [Majumder et al. 2007], Linguistica [Goldsmith 2001], and Oard [Oard et al. 2001]), and the baseline strategy that did not use stemming.

The organization of the article is as follows. In Section 2, we review some of the related existing work on statistical stemming. The proposed method is presented in Section 3. In Section 4 we provide the experimental setup, description of data, and the weighting formula used for retrieval. The effect of different parameter settings is discussed in Section 4.3. Results of retrieval experiments are presented in Section 5. Stemmer strength and error analysis are given in Section 6 and 7, respectively. The efficiency of our algorithm is discussed in Section 8. We conclude in Section 9.

2. RELATED WORK

Methods for handling morphological variation in information retrieval may be classified into reductive methods and generative methods [Kettunen 2009]. In the latter methods, morphological variation is not controlled when constructing the database index but only at query time by expanding queries to cover the morphological variants of the search keys. Among the former methods, stemming and lemmatization are the main approaches. In the present paper, we focus on stemming methods, and statistical stemming, in particular. Statistical stemming is an effective and popular approach in information retrieval [Xu and Croft 1998; Oard et al. 2001]. Some recent studies [Majumder et al. 2007; Bacchin et al. 2005] show that statistical stemmers are good alternatives to rule-based stemmers. Additionally, their advantage lies in the fact that they do not require language expertise. There are mainly three kinds of approaches to language-independent morphological normalization in information retrieval. Most methods take a set of words and try to find probable stems and suffixes for each word; other methods look for association between lexicographically-similar words by analyzing their co-occurrence in a corpus. The third group of methods is based on character *n*-grams. In this section, we review these approaches.

Majumder et al. [2007] developed a language-independent, clustering-based unsupervised technique (YASS) for word normalization capable of handling a family of primarily suffixing languages. They defined a set of string distance measures between word pairs that reward long-matching prefixes and penalize an early mismatch. Complete linkage clustering was used to discover groups of (presumably morphologically related) words. They conclude that in terms of retrieval effectiveness, this approach is comparable to rule-based stemmers like Porter or Lovins for English. For Bengali and French, this approach provides substantially improved performance as compared to using no stemming.

Goldsmith [2001] presented an unsupervised morphological analyzer based on minimum-description-length model. The frequency of stems and suffixes that result from every possible breakpoint in each term of a collection is examined. A breakpoint for each token is optimal if every instance of a token must have the same breakpoint and which minimizes the number of bits necessary to encode the collection. The underlying intuition is that breakpoints should be chosen in such a way that each token can be segmented into relatively common stems and common suffixes. Linguistica is the available software based on this idea.

Bacchin et al. [2005] described a language-independent probabilistic model for stemmer generation which makes use of the mutual reinforcement relationship between stems and suffixes. From a finite collection of words, they first generated a set of substrings (prefixes and suffixes) by splitting each word at all possible positions, except for those which generate empty substrings. Then they form a directed graph where a node implies a substring and a directed edge between node x and y exists if there is a word z in the collection such that $z = xy$. Now they estimate the prefix score and suffix score using HITS algorithm with the assumption that the good prefixes point to good suffixes, and good suffixes are pointed to by good prefixes. Once the prefix and suffix scores are estimated, the algorithm determines the most probable split (into prefix and suffix pair) for each word in the collection which maximizes the probability of the prefix and suffix pair. A set of experiments with several languages (such as English, French, Italian, etc.) produced equally good results as those produced by Porter stemmer for these languages.

Oard et al. [2001] did suffix discovery statistically in a text collection and eliminated the word endings. They first counted the frequency of every one, two, three, and four character suffix that would result in a stem of three or more characters for the first 500,000 words of the collection. Then they subtracted the frequency of the most common subsuming suffix of the next longer length from each suffix (for example, frequency of “-ing” from the frequency of “-ng”). The adjusted frequencies were then used to sort all *n*-gram suffixes in descending order. In the case of English, the count versus rank plot was found to be convex and so the rank at which the second derivative was maximum was chosen as the cutoff limit for the number of suffixes for each length.

Xu and Croft [1998] investigated the usefulness of corpus analysis in ad hoc retrieval. The basic assumption of their work is that word variants that should be conflated will occur in the same documents or, more specifically, in the same text window (they reported on 100 word text window). Based on this assumption, they devise a co-occurrence metric that measures the association of two words. Graph-partitioning algorithms (such as connected component and optimal partitioning) are then used to refine the initial classes generated by different aggressive stemmers like Porter or Krovetz. This technique helps to identify the corpus-dependent conflations and also minimizes the bad conflations (universal and university to the same class) made by linguistic rule-based stemmers. The reported results show that corpus analysis, indeed, improved the performance significantly on both English and Spanish corpora.

Character *n*-gram tokenization [Mcnamee and Mayfield 2004] is another attractive option to handle morphology in an alphabetic language. It was reported that 4-grams, in particular, perform well for European languages. One major pitfall with *n*-grams is the increase in the size of the inverted index. A passage of k characters contains $k - n + 1$ *n*-grams of length n, but only approximately $(k + 1)/(l + 1)$ words, where l is the average word length for the language. As a consequence (of index size expansion), there is a hefty increase in query processing time when *n*-grams are used: retrieval with 4-grams is ten times slower than plain word retrieval.

3. PROPOSED APPROACH: GRAS

3.1. Terminology

For the rest of the article we define the following terms.

—*Co-occurring Suffix Pair*. Let L be the lexicon of words contained in a corpus. A pair of suffixes $\langle s_1, s_2\rangle$ is said to be *co-occurring* if there exists a pair of distinct words $\langle w_i, w_j\rangle$ in L, such that $w_i = rs_1$, $w_j = rs_2$, where r is the longest common prefix of w_i and w_j, and $|r| > 0$. Informally, s_1 and s_2 are regarded as a co-occurring pair if each of s_1 and s_2 may be added as a suffix to the end of a common "*root*" r to form valid words w_i and w_j. The pair $\langle w_i, w_j\rangle$ is said to induce the suffix pair $\langle s_1, s_2\rangle$, and the number of such word pairs is termed the frequency of the suffix pair $\langle s_1, s_2\rangle$.

For example, if $L = \{activate, activation, educate, education\}$, the suffix pair $\langle e, ion\rangle$ is said to co-occur with frequency 2. Note that at most one of $\langle s_1, s_2\rangle$ is permitted to be null; in this case one word is a substring of the other. For example, $\langle neutral, neutralize\rangle$ gives rise to the suffix pair $\langle NULL, ize\rangle$.

—*α-frequent Suffix Pair*. A pair of co-occurring suffixes $\langle s_1, s_2\rangle$ is said to be *α-frequent* if its frequency $\geq \alpha$, that is, if there are at least α strings which take both s_1 and s_2 as suffixes.

—*Neighbour*. Two distinct words $\langle w_1, w_2\rangle$ are said to be *neighbours* of each other if $w_1 = rs_1$, $w_2 = rs_2$ (r is the non-zero length longest common prefix of $\langle w_1, w_2\rangle$), and the suffix pair $\langle s_1, s_2\rangle$ is *α-frequent*.

3.2. Overview

Consider the following problem: we are given a set of distinct words and our goal is to partition the words into a set of classes, where each class represents a set of morphologically related words. Given that we are considering only suffixing languages, the class formation can be achieved easily and accurately if we have access to the linguistically valid suffixes. This requires language-specific knowledge.

Since our goal is to develop a language-independent stemming algorithm, we do not use any list of linguistically valid suffixes, nor do we employ any complex methodology

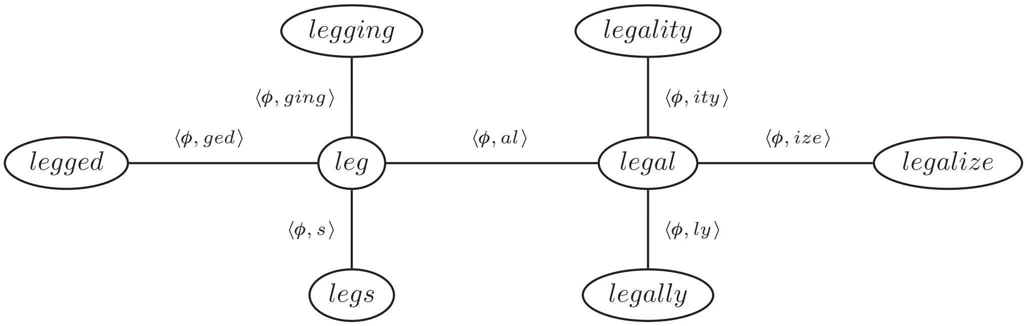

Fig. 1. Illustration of Principle (2) ϕ stands for NULL suffix.

to identify linguistically valid suffix pairs. Instead, we automatically identify a set of co-occurring suffix pairs along with their frequencies from the lexicon (described in section 3.3). The identified suffix pairs are then used to establish the relationship between word pairs. The following two principles for a class of morphologically related words are the key to the proposed algorithm.

(1) There is a specially designated word called the *pivot* (central word) in each class and the other words in the class are neighbours of the pivot.
(2) For every word in a class (except the pivot), most of its neighbors are also neighbors of the pivot for that class.

The role of Principle (2) is very important for the functioning of the algorithm. For example, the English suffix pair $\langle NULL, al\rangle$ is very frequent and connects many morphologically related word pairs such as $\langle accident, accidental\rangle$, $\langle aspiration, aspirational\rangle$. But the same suffix pair should not be used to connect the word pair $\langle leg, legal\rangle$, since these are formed from different roots. Thus, while using Principle (1) alone may, in many instances, group together morphologically unrelated words, Principle (2) prevents such incorrect groupings by incorporating the neighbourhood information. More specifically, Principle (2) ensures that *leg* and *legal* would be regarded as morphological variations of the same root only if *leg* and *legal* have a number of common neighbours. Figure 1 shows this example graphically.

The use of Principle (2) enables GRAS to effectively resolve some errors committed by other methods such, as those proposed by Oard et al., and YASS. For example, in Oard's approach, "*-al*" is likely to be identified as a suffix and will therefore be removed from the end of the word *legal*. The proposed algorithm makes use of Principle (2) to prevent such errors.

GRAS is also able to avoid certain errors made by YASS. For example, Majumder et al. mention the words *akram* (a proper noun) and *akraman* (attack), which were stemmed to the same form by YASS. These were correctly grouped into separate classes by GRAS.

Indeed, a rule-based algorithm such as Porter's also stems certain words incorrectly. For example, the words {*execute, executive*} are stemmed to *execut*, and {*illegible, illegal*} are stemmed to *illeg*) by Porter's stemmer. Once again, these are handled correctly by GRAS through the application of Principle (2).

3.3. Suffix Pair Identification

Algorithm 1 describes a method for the identification of cooccurring suffix pairs. We start with a lexicon containing the distinct words of a corpus. The words are then partitioned into a number of groups such that each pair of words drawn from a group has a longest common prefix of length at least a given threshold value ℓ. This step can

ALGORITHM 1: Identify Suffix Pairs

```
Let $S = \{C_1, C_2, \ldots, C_n\}$ be the set of classes (of words) such that any two words in any
  particular class have a longest common prefix of length $\geq \ell$.
for $i = 1$ to $n$ do
   Let $C_i = \{w_1, w_2, \ldots, w_m\}$.
   for any two words $w_j, w_k \in C_i$ $(j < k)$ do
      Output the suffix pair $\langle s_1, s_2 \rangle$ such that $w_j = rs_1$, $w_k = rs_2$, and $r$ is the longest common
        prefix of $\langle w_j, w_k \rangle$.
   end for
end for
Compute the frequency of all suffix pairs.
```

be done in a single linear pass over the sorted lexicon. Each group is now considered in turn, and the suffix pairs are derived from each pair of words from that particular group. Although theoretically this step takes quadratic time (in the size of each group), in practice it consumes only a few seconds, since the size of the common prefix sharing word classes are small even for a moderately low value of common prefix length. When all groups are exhausted, the frequencies of suffix pairs are computed. Finally a set of α-*frequent* suffix pairs is constructed. For the actual implementation, we find the group of common prefix-sharing words in a linear pass over the sorted lexicon. Then for each group we consider every pair of words of that group and collect the suffix pair by truncating their longest common prefix.

The choice of ℓ and α is important for the performance of the algorithm. We observed that word pairs sharing longer prefixes are more likely to be morphologically related than pairs that share shorter prefixes. Suffix pairs derived from long prefix-sharing word pairs are therefore more likely to be linguistically valid. Thus, a low value of ℓ can be a bad choice as it will lead to the identification of many invalid suffix pairs and a high value may miss out on some valid but infrequent suffix pairs. Therefore, as a balancing act, we set the value of ℓ to be the average word length for the language concerned. The other parameter, namely the suffix frequency cut-off (α), is used to prune invalid (or irregular) suffix pairs. Removing all and only the invalid suffix pairs using only frequency information is a hard task. We did not choose a high value of α, since that would cause rejection of many valid suffix pairs. A low value of α is safer. Even though this may lead to the identification of invalid co-occurring pairs, the use of Principle (2) prevents unrelated words from being grouped together in the same class. We present the experimental results on various cut off values in Section 4.3.

3.4. Class Formation

We use the information computed by Algorithm 1 to construct a weighted undirected graph $G = (V, E)$ as follows. Each vertex of G represents a word in a given lexicon L. Let $u, v \in V$ be two words from the lexicon L, and let $w(u, v)$ be the frequency of the suffix pair induced by the word pair represented by the vertices u and v. If $w(u, v) \geq \alpha$ (a cut-off value), then we connect u, v with an edge, and assign the weight $w(u, v)$ to this edge. Thus $E = \{(u, v) \mid w(u, v) \geq \alpha\}$. Clearly, since $w(u, v) = w(v, u)$, G is undirected. The degree of a vertex means the number of edges incident on it, independent of the edge weight, and the term *Adjacent*(v) represents the set of nodes which have an edge from v. Note that v does not belong to *Adjacent*(v).

Once the graph G is constructed, Algorithm 2 iteratively finds the set of classes by decomposing the graph. The decomposition process proceeds as follows. It first chooses a pivotal node (say p) from the remaining vertices, such that its degree is maximum (ties are broken based on the lexicographical order). The underlying rationale is that if

ALGORITHM 2: Identify Class

```
while $G \neq \phi$ do
  Let $u$ be the vertex of $G$ with maximum degree.
  $S = \{u\}$
  for all $v \in Adjacent(u)$ taken in decreasing order of $w(u, v)$ do
    if $cohesion(u, v) \geq \delta$ then
      $S = S \cup \{v\}$
    else
      Delete the edge $(u, v)$.
    end if
  end for
  Output the class $S$.
  From $G$ remove the vertices in $S$ and their incident edges.
  Let $G^{'}$ be the new graph.
  $G = G^{'}$
end while
```

a word has many neighbours it is most likely a potential root, and a class can be formed treating this vertex as the central word. Algorithm 2 then considers the vertices adjacent to the pivotal node p in descending order of the edge weight $w(p, v)$ (i.e. the most potential neighbour first), and measures the cohesion between p and v using the formula.

$$cohesion(p, v) = \frac{1 + |Adjacent(p) \cap Adjacent(v)|}{|Adjacent(v)|}. \quad (1)$$

Clearly, the value of *cohesion* lies between 0 and 1 and a higher value of *cohesion* suggests a higher likelihood that the two nodes are morphologically related. If the *cohesion* value exceeds a certain threshold (say δ), the vertex v is assumed to be morphologically related with the pivot p and is put in the same class as p. Otherwise, the edge (p, v) is deleted immediately to mark that p and v are not related and the vertex v is kept intact to be considered as a possible member of another class. Therefore, each iteration identifies a class and removes it from the graph. The algorithm terminates when there are no more vertices to process.

Choice of δ A value of δ higher than 0.5 implies that more than half of the neighbours of a node v are shared by the pivotal node for the class containing v. Setting a value very close to 1 pays too much attention to every neighbour of the candidate node (the node that will be considered for grouping with the pivotal node), whereas a smaller value (less than 0.5) ignores more than half of the neighbours of v. To keep a balance between these two situations, a value higher than 0.5 and smaller than 1.0 may be a good choice. We provide the experimental results obtained by using several δ in Section 4.3.

4. EXPERIMENTAL SETUP

In order to assess the effectiveness of the approach proposed in Section 3, we tested our method along with other existing methods on a number of standard test collections. The experimental methodology followed for each language is summarized here.

(1) From the corpus, collect all unique words after removing the stopwords and numbers.
(2) Find the co-occuring suffix pairs and their frequencies (Algorithm 1).
(3) Generate the classes using Algorithm 2.
(4) Perform retrieval runs using the classes generated by GRAS for stemming.
(5) Compare the retrieval results against other strategies (no stemming, rule based, Oard's approach, Linguistica, and YASS).

We compared the performance of GRAS with no stemming (baseline) and four other word-form normalizers, namely, YASS, Linguistica, Oard's approach, and one rule-based stemmer for each language. We wanted to see how our approach, which is computationally inexpensive, performs compared to the stemmers based on language specific rules and other more computation-intensive but effective methods like YASS[1] and Linguistica,[2] as well as a relatively simple approach like Oard's. The English rule-based stemmer is the Porter stemmer, and the French rule-based stemmer is the French version of Porter stemmer. For Marathi and Bengali, we used the stemmer described in Dolamic and Savoy [2010]. For Hungarian and Czech, we used the stemmers presented in Savoy [2008] and Dolamic and Savoy [2009], respectively.[3] All of the rule-based stemmers mentioned above function like Porter stemmer: they remove manually identified suffixes from the ends of words based on language specific rules.

Among the language-independent methods, Linguistica appears to be particularly computation-intensive. When the system was run on the Czech lexicon containing 457,164 unique words, it could not finish the task even after more than seven hours. Therefore, the lexicon of each language was split into groups of 100,000 words and processed using Linguistica (the article by Goldsmith [2001] reports respectable performance using as few as 5,000 words).

4.1. Experimental System and Weighting Formula

To perform retrieval experiments, we used the TERRIER[4] information retrieval system. TERRIER implements a number of divergence-from-randomness based weighting schemes along with other well-known formula such as tf-idf and the language modeling approach. Throughout our experiments, we used the IFB2 model [Amati and Van Rijsbergen 2002] for term weighting. The IFB2 model fixes a term's weight in a particular document by multiplying the tf-ictf (term frequency-inverse collection term frequency) with a factor (given in Equation (3)), which is the ratio between the average elite set (the set of documents which contains the term) term frequency and the document term frequency. In other words, it boosts up (or keeps unchanged) a term's tf-ictf weight if the average elite set term frequency is higher than (or equal to) the term frequency in that particular document. The within document term frequency is also normalized based on the document length as shown in Equation (5). Equations (2) to (5) represent the ranking function and the weighting formula for the IFB2 model.

$$Score(Q, D) = \sum_{t \in Q} query\_term\_freq(t) \cdot f_1^D(t) \cdot f_2^D(t) \tag{2}$$

$$f_1^D(t) = \frac{collection\_term\_freq + 1}{df \cdot (tfn + 1)} \tag{3}$$

$$f_2^D(t) = tfn \cdot log_2 \frac{collection\_size + 1}{collection\_term\_freq + 0.5} \tag{4}$$

$$tfn^D(t) = tf \cdot log_2 \left(1 + \frac{avg\_doc\_length}{doc\_length}\right). \tag{5}$$

[1] http://www.isical.ac.in/∼clia/resources.html.

[2] http://linguistica.uchicago.edu/.

[3] For all languages except English, we used the programs for rule-based stemmers available at http://members.unine.ch/jacques.savoy/clef/index.html.

[4] http://ir.dcs.gla.ac.uk/terrier/.

Table I. Statistics on Test Corpora

Corpus →	Marathi	Hungarian	English	Czech	Bulgarian	Bengali	French
No. of documents	99,362	49,530	472,525	81,735	87,281	123,047	177,452
No. of unique words	854,324	528,315	522,381	457,164	320,673	533,605	303,349
Mean words per/doc	273	173	284	242	153	362	228
No. of queries	50	98	150	50	100	50	100
No. of rel. doc	621	2,219	12,848	762	2,261	510	4,685

4.2. Description of Data

We carried out retrieval experiments on Marathi, Hungarian, Czech, English, Bulgarian, Bengali, and French. The Marathi and Bengali collections are a part of the FIRE[5] test collections and comprise newspaper articles. We perform retrieval experiments on Marathi and Bengali using the topic sets taken from FIRE 2010. There are 50 topics for each of Marathi and Bengali. From the Marathi set we removed 11 topics for which no relevant documents exist. The English data consists of the FBIS, Financial Times, and LA Times collection from TIPSTER disks 4-5 along with 150 TREC topics from 301 to 450. The Hungarian, Czech, Bulgarian, and French collections are taken from CLEF[6] evaluation campaign. We took 98 topics from CLEF 2006 and 2007 for Hungarian (2 topics were removed as they had no relevant documents), 50 topics for Czech (CLEF 2007), 100 topics for Bulgarian (CLEF 2006 and 2007), and 100 topics for French (CLEF 2005 and CLEF 2006). In Table I, we summarize statistics about the test data sets used in our experiments.

4.3. Parameter Setting

In this section, we study the performance of GRAS under different parameter settings. To reduce the bias in learning parameters, we used different query sets for training (this Section) and testing (Section 5). For each of the languages—Hungarian, Marathi, Bulgarian, Bengali, and French—the training set consists of 50 queries (taken from CLEF 05, FIRE 08, CLEF 05, FIRE 08, and CLEF 04, respectively) while the English training set contains 200 queries (TREC topics 1-200 on WSJ test collection). We did not train on Czech because it had only 50 topics. The parameter values are learnt from the training queries based on the Mean Average Precision (MAP) values and the final results on test queries are presented under the learnt parameter settings. While experimenting on training queries, we particularly address the following questions.

(1) Are the parameters (α and δ) language independent?
(2) What are the good values of α and δ for the purpose of retrieval?
(3) Are α and δ mutually independent or do they influence each other?

For all the above questions, inferences are drawn based on the MAP values. We start with some values of the parameters which are somewhat intuitively justified and then carry out retrieval experiments across a spectrum of values.

Table II shows the MAP values obtained on six training query sets under different combinations of α and δ. Clearly, a value of δ between 0.8 and 0.9 provides consistently good performance across all languages. For Bulgarian, English, and French, the best performance is achieved at δ=1.0, but for other languages, the performance goes down at $\delta = 1.0$. On the other hand, for Marathi, the best result is obtained at relatively low values of δ compared to other languages.

Like δ, α also has some effect on the performance. A value of α between 4 to 8 seems most suitable for most of the languages. Bengali did not show much sensitivity to α, but

[5] http://www.isical.ac.in/∼clia/.

[6] http://www.clef-campaign.org/.

Table II. Effect of α and δ on Training Queries (MAP values)

$\delta \rightarrow$	$\alpha \downarrow$	0.5	0.6	0.7	0.8	0.9	1.0
Hun	2	0.334	0.328	0.331	0.331	0.331	0.330
	4	0.327	0.332	0.333	**0.343**	0.333	0.326
	6	0.331	0.337	0.337	0.335	0.331	0.333
	8	0.333	0.338	0.337	0.334	0.336	0.312
	10	0.337	0.335	0.339	0.334	0.335	0.297
Mar	2	0.410	0.411	0.416	0.399	0.401	0.388
	4	0.410	0.417	0.416	0.411	0.397	0.379
	6	**0.421**	0.420	0.404	0.402	0.407	0.391
	8	0.415	0.403	0.403	0.402	0.410	0.379
	10	0.406	0.403	0.399	0.413	0.410	0.373
Eng	2	0.273	0.284	0.289	0.293	0.293	0.295
	4	0.282	0.287	0.292	0.297	0.297	**0.298**
	6	0.285	0.288	0.294	0.296	0.293	0.293
	8	0.285	0.291	0.292	0.293	0.290	0.293
	10	0.285	0.291	0.292	0.295	0.292	0.291
Bul	2	0.253	0.256	0.259	0.261	0.265	0.247
	4	0.249	0.254	0.264	0.271	0.271	**0.274**
	6	0.257	0.257	0.271	0.277	0.269	0.272
	8	0.259	0.264	0.264	0.276	0.278	0.281
	10	0.250	0.260	0.267	0.272	0.273	0.267
Ben	2	0.372	0.369	0.380	0.379	0.382	0.380
	4	0.384	0.380	0.383	0.381	0.389	0.386
	6	0.379	0.384	0.388	0.379	**0.391**	0.389
	8	0.380	0.387	0.384	0.383	0.386	0.384
	10	0.386	0.388	0.385	0.386	0.387	0.384
Fre	2	0.319	0.317	0.313	0.328	0.334	0.345
	4	0.318	0.313	0.323	0.341	0.343	0.341
	6	0.309	0.313	0.329	0.340	0.337	0.345
	8	0.315	0.317	0.325	0.339	0.345	**0.349**
	10	0.314	0.321	0.333	0.337	0.334	0.331

English and French were found to be sensitive to some extent. A general observation that comes out of Table II is that for all languages except Marathi, higher values of δ (typically between 0.8–0.9) work well.

Based on Table II, we recommend the following values: $\alpha = 4, 6, 8$ and $\delta = 0.8$–0.9. In order to study the robustness of these settings, we apply paired t-tests on obtained results. We found that for all languages, the best performance figures were not significantly better than those obtained by setting $\alpha = 4$ and $\delta = 0.8$. Therefore, for all subsequent experimental results, we use these values of α and δ.

5. EXPERIMENTAL RESULTS

In this section we present the retrieval performance of the proposed algorithm along with a comparative study with rule-based stemmers and three other language-independent approaches. For each language, we present two tables: one showing MAP value, R-precision, prec@5, prec@10, and the number of retrieved relevant documents (when 1,000 documents are retrieved per query). The other table shows the query-wise performance comparison and the p value of the statistical significance test. Paired t-tests (on the basis of the per query average precision values) were used to infer whether one method is statistically significantly superior to another. Throughout the result section a p-value less than 0.05 (a confidence level of 95%) indicates the superiority of one method against other. We also provide a recall-precision curve for each language. To better understand the query wise performance, we present a figure (for each language) depicting the performance of the proposed algorithm against the method which achieves the closest MAP to GRAS. In each table, the quantities in bold indicate the best performance in that particular category. The queries are formed using

Table III. Retrieval Results for Hungarian

	MAP	R-PREC	P@5	P@10	Rel.Ret
WORD	0.239	0.252	0.357	0.314	1367
LING	0.295(23.8)	0.310(23.0)	0.422(18.3)	0.358(14.0)	1738(27.1)
OARD	0.267(11.9)	0.285(13.3)	0.400(12.0)	0.340(8.1)	1546(13.1)
YASS	0.306(28.2)	0.315(25.2)	0.449(25.7)	0.384(22.1)	1730(26.6)
RULE	0.313(31.3)	0.312(23.8)	0.445(24.6)	0.399(26.9)	1723(26.0)
GRAS	**0.351**(46.8)	**0.360**(43.1)	**0.474**(32.6)	**0.422**(34.4)	**1924**(40.7)

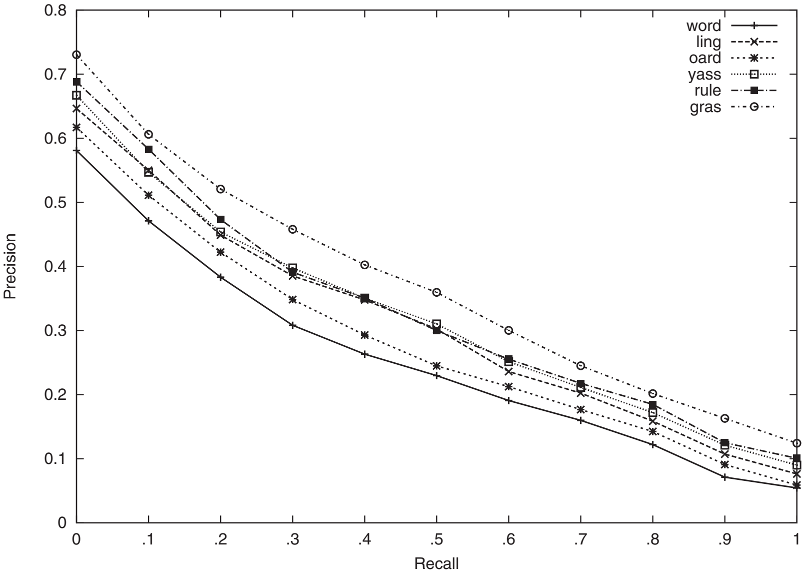

Fig. 2. Precision recall curve for Hungarian.

the title and description fields of the test topics. The quantity within parentheses in each table shows the percentage of performance increase (or decrease) compared to unstemmed runs. The acronyms in the tables denote the following retrieval strategies: WORD = unstemmed words, LING = Linguistica, OARD = Oard's algorithm, RULE = rule based stemmer, and GRAS = the proposed approach.

5.1. Hungarian

Our first set of experiments investigates the performance of GRAS on Hungarian. For these experiments, we conducted five runs. Table III presents the results for Hungarian using different stemmers. GRAS provides the best MAP across all methods studied and gives an improvement of 46.8% relative to the unstemmed word-based retrieval. The rule-based stemmer performs better than YASS, Linguistica, and Oard but not better than GRAS. YASS and Linguistica perform almost equally, with YASS having a marginal edge over Linguistica, and both these methods perform quite well compared to the unstemmed run. OARD only managed to provide an improvement of 11.9% in MAP compared to WORD.

The interpolated precision achieved by the six methods at each recall point is shown in Figure 2. GRAS, YASS, RULE, and Linguistica maintain a consistent performance improvement over no stemming. GRAS performs better than the other methods in terms of precision at different recall levels, as well as the number of relevant documents retrieved.

Table IV. Query-by-Query Analysis and Significance Test for Hungarian

	LING		OARD		YASS		RULE	
	poorer	better	poorer	better	poorer	better	poorer	better
# queries	67(50)	30(12)	76(65)	20(10)	66(46)	31(13)	63(52)	33(23)
p-value	0.75×10^{-5}		1.65×10^{-8}		1.48×10^{-5}		0.0003	

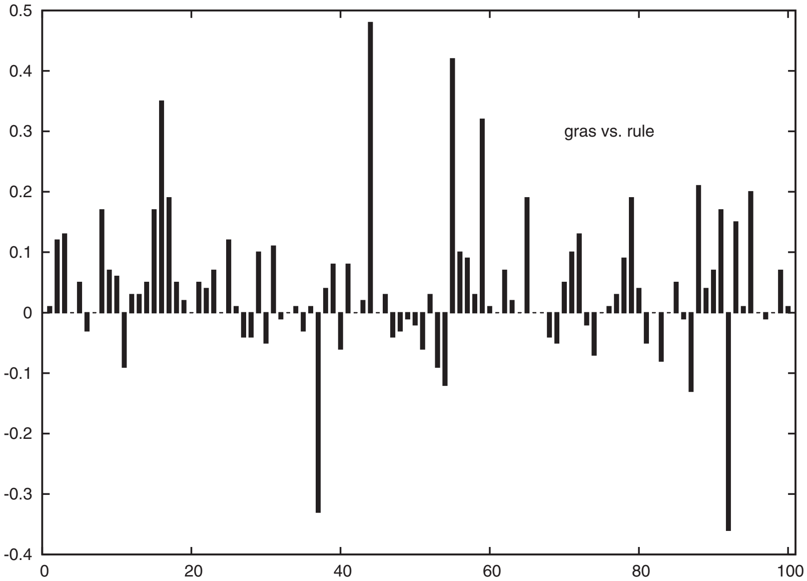

Fig. 3. Query-by-query graph on Hungarian.

Table IV compares the query-specific performance of each method. The quantities in the table denote the number of queries for which each method yields a higher/lower AP than GRAS. The quantities in parentheses show the number of queries for which the performance difference is 10% or more. For example, GRAS achieved a better AP than YASS and RULE on 66 and 63 queries, respectively, and the relative difference was at least 10% on 46 and 52 queries respectively. On the other hand, YASS and RULE performed better on 31 and 33 queries, respectively. Figure 3 shows the query-wise performance comparison between GRAS and RULE. The vertical bars above (or below) the x-axis represent that GRAS is better than RULE (or RULE is better than GRAS). Table IV (the row corresponding to p-value) presents the results of statistical significance tests between GRAS and each of the other normalizers separately. Clearly, GRAS is very significantly more effective compared to LING, OARD, YASS, and RULE.

5.2. Marathi

Marathi is an Indian language with a rich morphology. Once again, GRAS is the top performer with a relative performance improvement of 43.4% compared to the retrieval using unstemmed words. Linguistica and YASS also did fairly well in terms of both MAP and the number of relevant documents retrieved. However, Oard's approach only managed to provide an improvement of 4% against no stemming. Interestingly, RULE performed worse than all other methods except OARD and provided very marginal improvement compared with WORD. Table V summarizes the results for the 6 methods. The graph in Figure 4 shows that GRAS produces a better precision at each recall level than any other method. The performance difference between GRAS and each of the other methods was found to be statistically significant.

Table V. Retrieval Results for Marathi

	MAP	R-PREC	P@5	P@10	Rel.Ret
WORD	0.315	0.315	0.364	0.356	551
LING	0.411(30.7)	0.393(24.9)	0.415(14.1)	0.380(6.5)	590(7.1)
OARD	0.327(4.0)	0.330(4.7)	0.369(1.4)	0.336(−5.8)	561(1.8)
YASS	0.372(18.1)	0.347(10.2)	0.436(19.7)	0.377(5.8)	589(6.9)
RULE	0.329(4.5)	0.313(−0.7)	0.374(2.8)	0.367(2.9)	555(0.7)
GRAS	0.451(43.4)	0.430(36.6)	0.426(16.9)	0.380(6.5)	607(10.2)

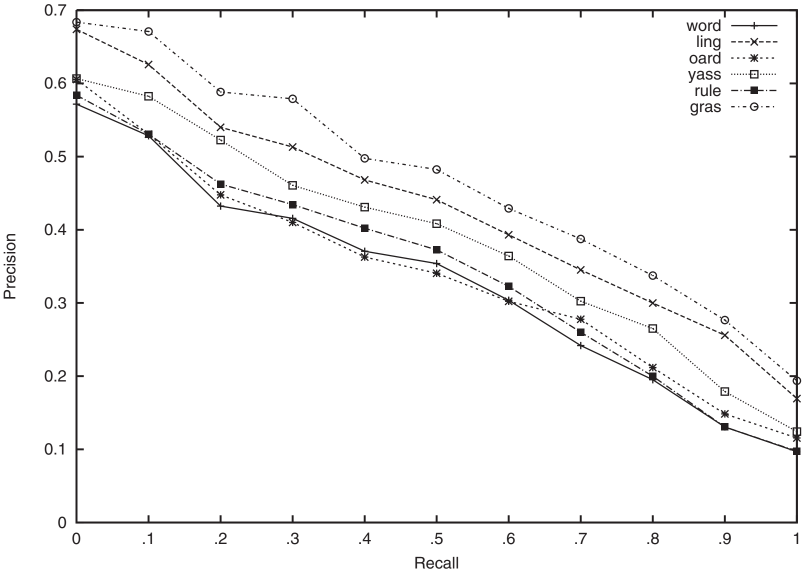

Fig. 4. Precision recall curve for Marathi.

In Figure 5 we show the query-by-query comparison between GRAS and Linguistica on 39 Marathi queries based on AP values. GRAS performed better than Linguistica for 23 queries. On 16 out of these 23 queries, GRAS performed at least 10% better than Linguistica, whereas Linguistica did at least 10% better than GRAS on 6 queries (Table VI). The comparison between GRAS and YASS reveals that GRAS gives better performance on 25 queries with 15 queries at least 10% better.

5.3. Czech

In this section we report the experimental results on the Czech data set. Table VII shows that all the strategies (rule-based and statistical) performed better than no stemming. Among the strategies, GRAS is the best. GRAS gives a relative performance improvement of 53.8% in MAP compared to no stemming, which is clearly very large and significant. Additionally, Table VII shows that normalization strategies improve the recall significantly. GRAS was able to retrieve, on an average, 2.7 more documents per query compared to no stemming.

Figure 6 depicts the same trend as the other languages: GRAS maintains a better (or at least equal) precision at all recall levels than the other methods. In Table VIII we present the results of statistical significance tests and query-by-query analysis of GRAS versus other methods. T-tests show that GRAS is significantly better than no stemming, Linguistica, Oard, YASS, and RULE. In Figure 7, we provide a query-by-query comparison between GRAS and YASS. Clearly, GRAS performs better than YASS on a substantial number of queries.

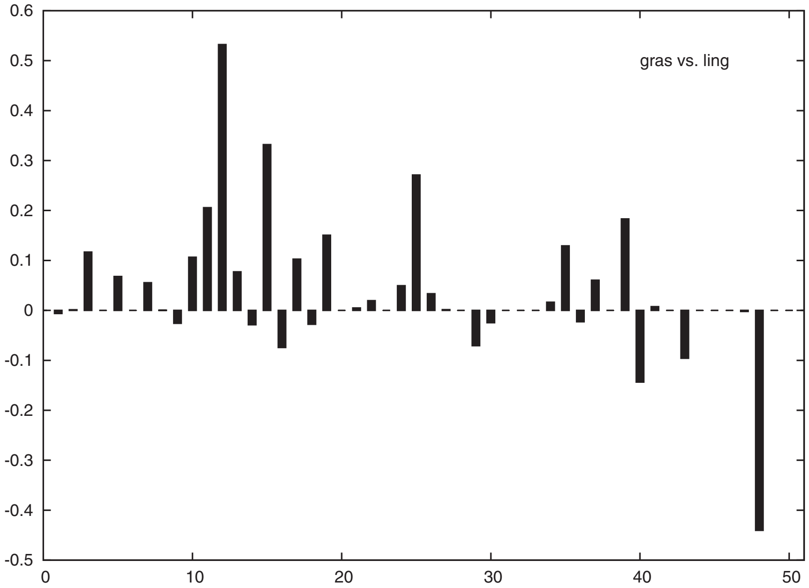

Fig. 5. Query-by-query graph on Marathi.

Table VI. Query-by-Query Analysis and Significance Test for Marathi

	LING		OARD		YASS		RULE	
	poorer	better	poorer	better	poorer	better	poorer	better
# queries	23(16)	12(6)	30(22)	7(5)	25(15)	12(7)	30(22)	8(6)
p-value	0.04		0.05×10^{-2}		0.015		0.003	

Table VII. Retrieval Results for Czech

	MAP	R-PREC	P@5	P@10	Rel.Ret
WORD	0.238	0.261	0.316	0.268	551
LING	0.308(29.5)	0.307(17.5)	0.392(24.1)	0.336(25.4)	664(20.5)
OARD	0.277(16.2)	0.286(9.7)	0.364(15.2)	0.298(11.2)	615(11.6)
YASS	0.342(43.4)	0.329(26.2)	0.448(41.8)	0.340(26.9)	672(22.0)
RULE	0.341(43.2)	0.346(32.4)	0.452(43.0)	0.348(29.9)	667(21.1)
GRAS	**0.366**(53.8)	**0.360**(37.9)	**0.448**(41.8)	**0.376**(40.3)	**684**(24.1)

5.4. English

In Table IX, we summarize the retrieval results obtained on the TREC collection for 150 queries (queries 301-450). Unlike Marathi, Hungarian, and Czech the performance differences of the five methods compared to no stemming are found to be much less. This is in agreement with the view that English is morphologically less complex than the other languages considered so far. Figure 8 also shows that the improvements in the precision at various recall levels are marginal compared to no stemming. Interestingly, Oard's stemmer, which performed relatively poorly on both the Marathi and Hungarian data sets, achieved nearly the same MAP with YASS and yielded better performance than Linguistica. Porter stemmer performed better than LING, OARD, and YASS but remained nearly 4% inferior to GRAS. Additionally, the Table IX shows that GRAS retrieved 122 more relevant documents than RULE.

Table X shows that GRAS performed better than LING, OARD, and YASS in a substantial number of queries. Although the number of queries in which GRAS and RULE beat each other nearly equal, GRAS achieved 10% better MAP in 32 queries, while

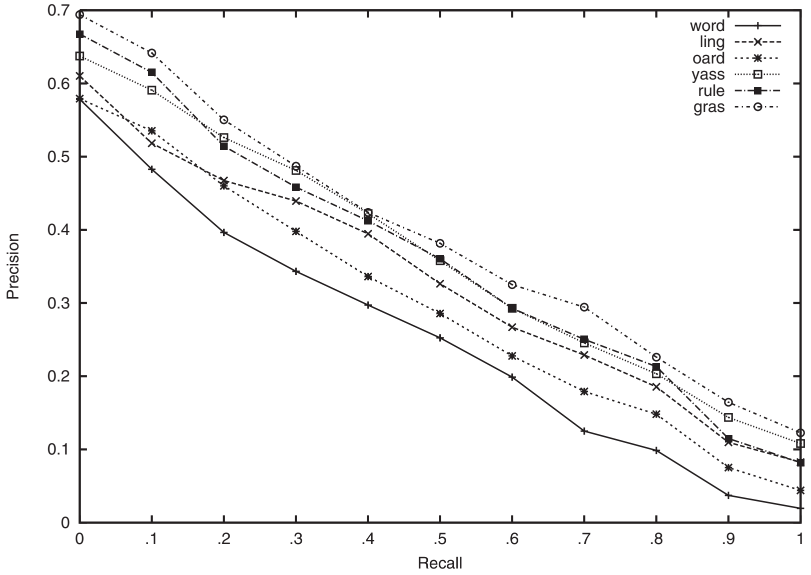

Fig. 6. Precision recall curve for Czech.

Table VIII. Query-by-Query Analysis and Significance Test for Czech

	LING		OARD		YASS		RULE	
	poorer	better	poorer	better	poorer	better	poorer	better
# queries	33(27)	17(8)	41(36)	9(7)	31(18)	17(11)	28(20)	20(12)
p-value	0.0004		3.40×10^{-5}		0.0162		0.048	

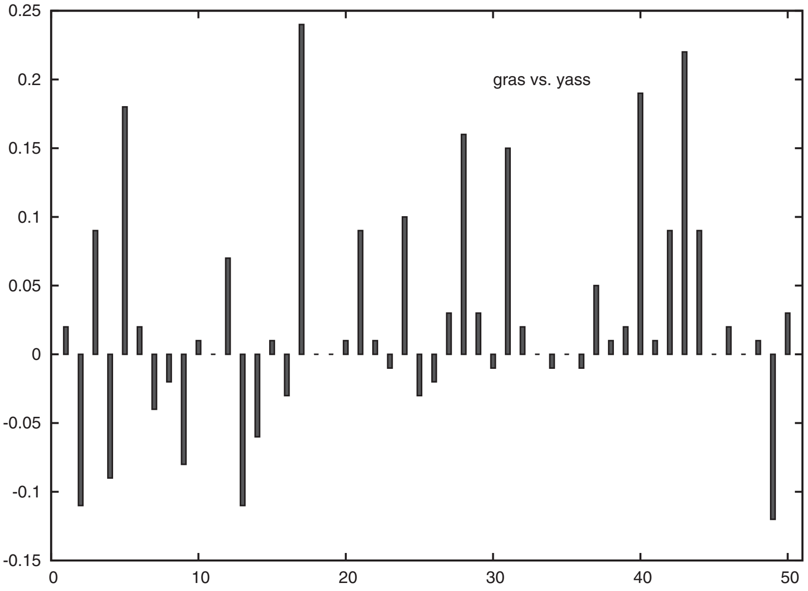

Fig. 7. Query-by-query graph on Czech.

Table IX. Retrieval Results for English

	MAP	R-PREC	P@5	P@10	Rel.Ret
WORD	0.229	0.273	0.500	0.433	6812
LING	0.236(3.0)	0.280(2.6)	0.505(1.1)	0.449(3.8)	7236(6.2)
OARD	0.251(9.6)	0.292(6.8)	0.508(1.6)	0.451(4.2)	7388(8.5)
YASS	0.250(9.1)	0.291(6.6)	0.527(5.3)	0.463(7.1)	7652(12.3)
RULE	0.260(13.5)	0.301(10.1)	0.529(5.9)	**0.483**(11.7)	7751(13.8)
GRAS	**0.270**(17.8)	**0.309**(13.0)	**0.543**(8.5)	0.479(10.8)	**7873**(15.6)

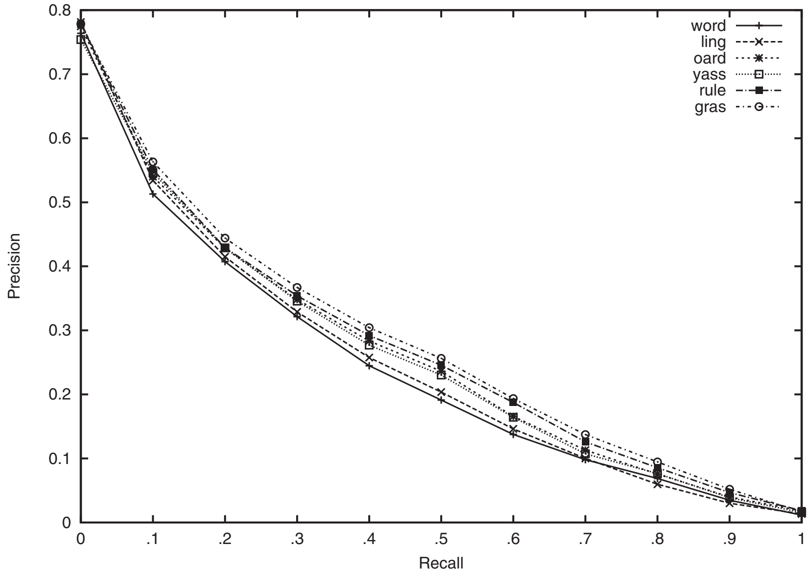

Fig. 8. Precision recall curve for English.

Table X. Query-by-Query Analysis and Significance Test for English

	LING		OARD		YASS		RULE	
	poorer	better	poorer	better	poorer	better	poorer	better
# queries	102(73)	46(26)	90(52)	58(26)	89(49)	59(32)	76(32)	71(22)
p-value	5.85×10^{-7}		0.001		0.001		0.036	

RULE did 10% better in 22 queries. Query-specific performance differences (GRAS vs. RULE) are shown in Figure 9. The inference drawn from the paired *t*-tests (table X) is that GRAS gives significant performance improvements over all the methods we have studied, including the Porter stemmer.

5.5. Bulgarian

The next experiment examines the retrieval effectiveness of stemming for Bulgarian. We take 100 queries from CLEF 2006 and 2007 for the experiment. Table XI shows that stemming methods clearly improve retrieval performance. The highest performance was achieved by GRAS which is 50.6% better than unstemmed word based indexing.

By analyzing the results (Table XI), we infer that Linguistica, RULE, and Oard's approach performed almost equally. YASS performed better than all strategies except GRAS. Unlike Hungarian and Marathi, the performance difference of GRAS with YASS is relatively small. However, GRAS is significantly better than no stemming, LING, OARD, YASS, and RULE. Figure 10 clearly demonstrates, once again, that GRAS improves precision at every recall point. The query-by-query analysis in Figure 11 and

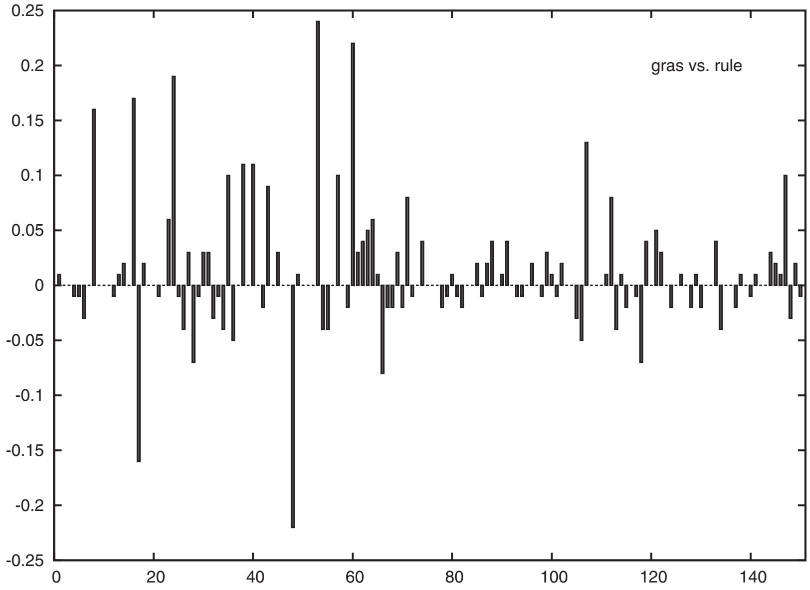

Fig. 9. Query-by-query graph on English.

Table XI. Retrieval Results for Bulgarian

	MAP	R-PREC	P@5	P@10	Rel.Ret
WORD	0.217	0.229	0.294	0.257	1611
LING	0.288(33.1)	0.293(27.6)	0.384(30.6)	0.321(24.9)	1973(22.5)
OARD	0.285(31.8)	0.297(29.4)	0.368(25.2)	0.338(31.5)	1944(20.7)
YASS	0.307(41.8)	0.316(37.9)	0.404(37.4)	0.348(35.4)	2085(29.4)
RULE	0.279(29.0)	0.293(27.8)	0.362(23.1)	0.327(27.2)	2003(24.3)
GRAS	**0.326**(50.6)	**0.334**(45.6)	**0.424**(44.2)	**0.355**(38.1)	**2110**(31.0)

Table XII indicates that GRAS outperforms YASS on a substantial number of queries. The performance differences are greater against Linguistica and OARD.

5.6. Bengali

The next set of experiments compares the performance of GRAS and those of the other strategies on the FIRE Bengali collection for 50 queries. Unlike Marathi, Hungarian, Czech, and Bulgarian, Bengali is less benefited due to stemming strategies. Yet it is found to be statistically significant compared to unstemmed word retrieval. Table XIII presents the performance figures of all the methods under various metrics. We can see in Table XIII that the relative improvement (to nostem) in MAP achieved by GRAS and YASS is more than 15%, with GRAS having almost 4% better than YASS. The performance difference between GRAS and RULE is even larger (almost 8.5%). But the performances of LING and OARD are much poorer. Note that although the number of relevant document retrieved by each of the methods is very close, still they differ significantly in MAP. This is because the methods (GRAS and YASS), which earn a better MAP, were able to retrieve more relevant documents at the top. The quantities (in Table XIII) corresponding to R-precision and P@5 ensure the observation.

Figure 13 depicts the query wise performance comparison of GRAS against YASS. GRAS and YASS are better than each other in 24 and 25 queries, respectively. However, the number of queries in which GRAS is better than other methods is larger. The recall-precision graph is given in Figure 12. Statistical significance tests (Table XIV) indicate

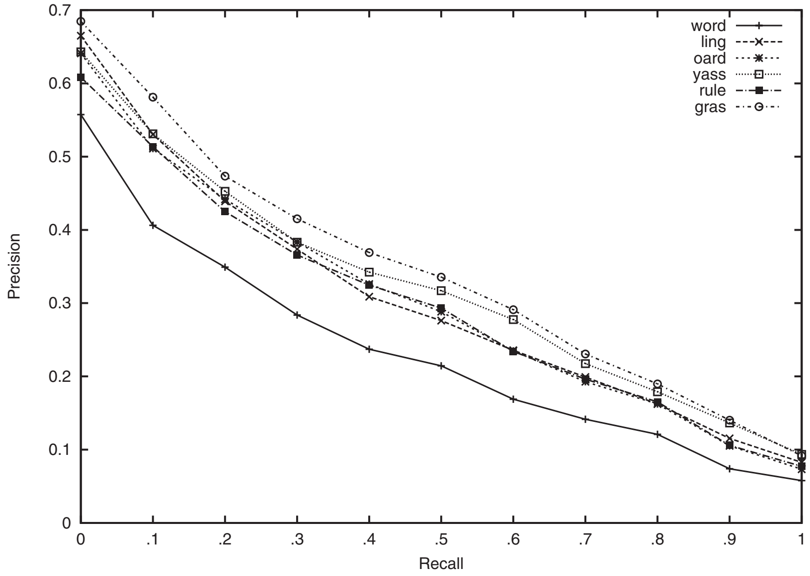

Fig. 10. Precision recall curve for Bulgarian.

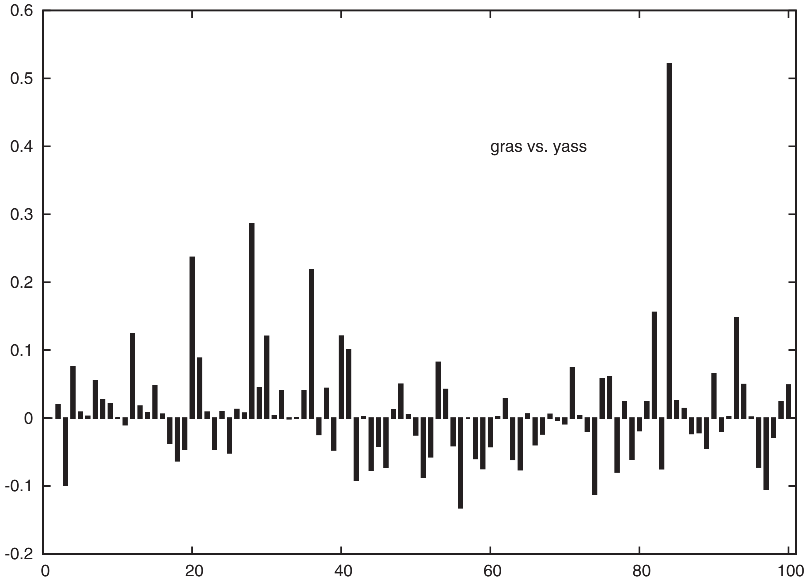

Fig. 11. Query-by-query graph on Bulgarian.

Table XII. Query-by-Query Analysis and Significance Test for Bulgarian

	LING		OARD		YASS		RULE	
	poorer	better	poorer	better	poorer	better	poorer	better
# queries	65(50)	34(16)	64(52)	36(18)	58(39)	41(22)	67(54)	33(22)
p-value	0.0007		0.0005		0.02		1.54×10^{-05}	

Table XIII. Retrieval Results for Bengali

	MAP	R-PREC	P@5	P@10	Rel.Ret
WORD	0.393	0.367	0.416	0.352	489
LING	0.419(6.6)	0.369(0.5)	0.396(−4.8)	0.334(−5.1)	**499**(2.0)
OARD	0.427(8.6)	0.406(10.5)	0.436(4.8)	0.344(−2.3)	495(1.2)
YASS	0.463(17.9)	0.415(13.0)	**0.476**(14.4)	**0.372**(5.7)	498(1.8)
RULE	0.443(12.6)	0.410(11.7)	0.428(2.9)	0.362(2.8)	**499**(2.0)
GRAS	**0.476**(21.0)	**0.435**(18.4)	0.464(11.5)	0.370(5.1)	**499**(2.0)

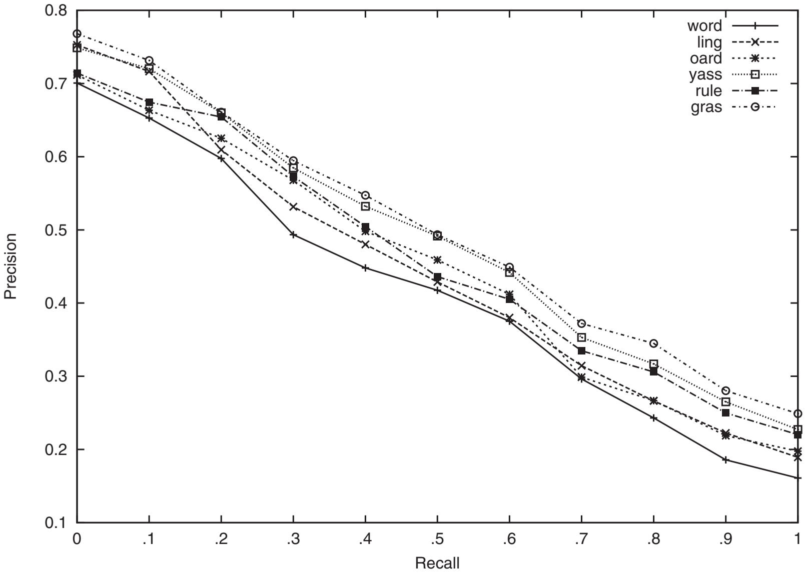

Fig. 12. Precision recall curve for Bengali.

that GRAS is significantly better than Linguistica, Oard, and RULE, but comparable to YASS.

5.7. French

As part of our last experiments, we study the effect of stemming on French data with 100 test queries taken from CLEF 05 and CLEF 06. Once again, GRAS outperforms the other methods in terms of MAP value. GRAS provides an improvement of nearly 14% MAP when compared with un-normalized retrieval. OARD and YASS performed almost equally, whereas RULE managed nearly 2% additional MAP improvement compared to either of OARD and YASS. On the other hand, GRAS is marginally better than RULE when MAP, precision at 5, and precision at 10 were considered, but RULE retrieved overall 24 more relevant documents than GRAS . Table XV gives the complete results.

Figure 14 depicts the precision recall curve for all the methods studied. Clearly, GRAS, RULE, YASS, and OARD are almost equally competent at various recall points, on the contrary, LING is slightly better than no stemming and its performance is poorer than the rest of the methods. A more detailed statistics based on the query wise AP values is given in Table XVI. GRAS remains superior to LING and YASS in a large number of queries and in almost twice the number of queries its performance was even 10% better. The result of statistical significance test is given in Table XVI. GRAS is significantly better than both LING and YASS, but the performance differences are not found significant when OARD and RULE were considered. Figure 15 also shows the query-specific performance between GRAS and RULE graphically.

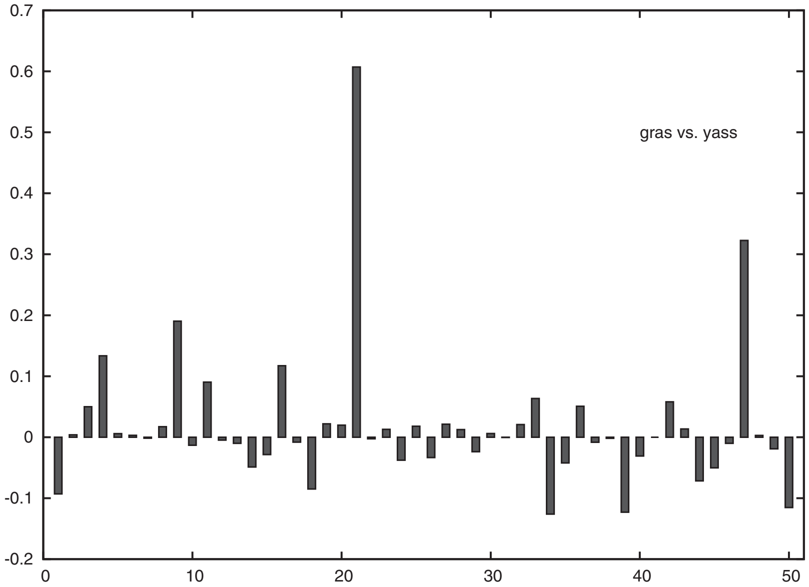

Fig. 13. Query-by-query graph on Bengali.

Table XIV. Query-by-Query Analysis and Significance Test for Bengali

	LING		OARD		YASS		RULE	
	poorer	better	poorer	better	poorer	better	poorer	better
# queries	28(17)	19(8)	30(18)	19(9)	24(11)	25(9)	29(17)	18(8)
p-value	0.008		0.036		0.14		0.04	

Table XV. Retrieval Results for French

	MAP	R-PREC	P@5	P@10	Rel.Ret
WORD	0.339	0.357	0.475	0.445	3796
LING	0.349(3.1)	0.364(1.8)	0.499(5.1)	0.458(2.9)	3990(5.1)
OARD	0.374(10.4)	0.385(8.0)	0.527(11.1)	0.483(8.4)	4061(7.0)
YASS	0.374(10.4)	0.389(9.0)	0.517(9.0)	0.473(6.1)	4055(6.8)
RULE	0.382(12.6)	**0.399**(11.8)	0.519(9.4)	0.487(9.3)	**4102**(8.1)
GRAS	**0.387**(14.3)	0.398(11.5)	**0.533**(12.3)	**0.491**(10.2)	4078(7.4)

Stemming is generally beneficial for improving performance of an IR system. Although it is known to be a recall enhancing device, nevertheless, its precision boosting power is noticeable for languages (such as Hungarian, Marathi, Czech) where the classes representing the morphological variations are of bigger sizes.

6. STEMMER STRENGTH

We now present a comparative study of various stemmers in terms of the stemmer strength. Stemmer strength generally represents the extent to which a stemming method changes words that it stems. One well-known measure of stemmer strength is the average number of words per conflation class [Frakes and Fox 2003]. Formally, if N_a, N_w, and N_s denote the mean number of words per conflation class, the number of distinct words before stemming and the number of unique stems after stemming respectively, then $N_a = \frac{N_w}{N_s}$. Table XVII gives the values of N_a for various stemming methods for each of the languages we have studied. Clearly, a higher value of N_a indicates a more aggressive stemmer.

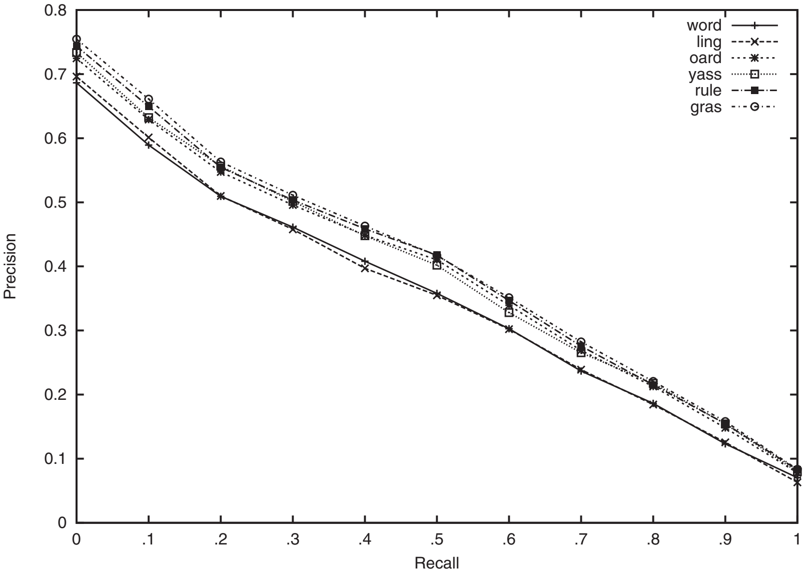

Fig. 14. Precision recall curve for French.

Table XVI. Query-by-Query Analysis and Significance Test for French

	LING		OARD		YASS		RULE	
	poorer	better	poorer	better	poorer	better	poorer	better
# queries	62(38)	35(17)	52(27)	42(19)	57(25)	39(14)	58(17)	36(16)
p-value	0.01×10^{-2}		0.14		0.04		0.25	

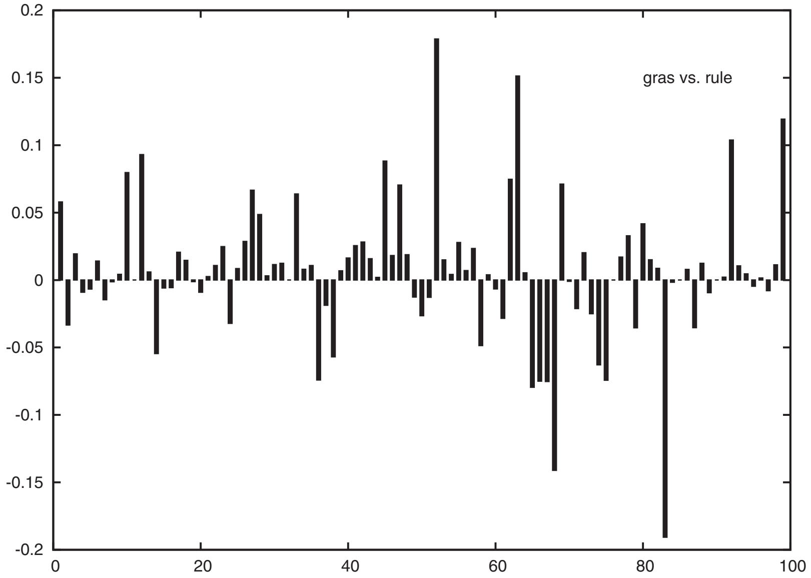

Fig. 15. Query-by-query graph on French.

Table XVII. Stemmer Strength

Language	LING	OARD	YASS	RULE	GRAS
Hun	1.99	1.21	3.81	2.05	3.97
Mar	2.38	1.20	3.30	1.15	3.44
Cze	2.71	1.45	4.09	2.80	3.29
Eng	1.61	1.17	2.73	1.23	1.03
Bul	2.30	1.51	3.72	1.65	2.93
Ben	2.06	1.24	3.29	2.26	2.67
Fre	1.43	1.22	3.01	1.60	1.73

Hungarian, Marathi, and Czech have larger conflation classes than the other languages. Bulgarian and Bengali also have relatively large classes. The table confirms that English and French are less inflectionally complex languages. Among the stemmers, YASS appears to be particularly aggressive on all languages, and produces the largest N_a values for Czech, English, Bulgarian, Bengali, and French. On the other hand, GRAS is the most aggressive on Hungarian and Marathi while it is about as strong as rule-based stemmers on English and French. OARD seems to be the lightest stemmer with a low value of N_a even for morphologically complex languages like Hungarian and Marathi.

7. ANALYSIS

In this section, we seek to gain further insights into our stemming algorithm by taking a closer look at a few individual queries for which the performance difference between GRAS and its closest rival is noticeable. We chose two languages we understand—namely, English and Bengali—for this analysis. Unfortunately, our ignorance of the other languages precludes the possibility of a similar analysis for these languages.

English. From Figure 9, we identify the queries 48 (TREC query no. 348) and 53 (TREC query no. 353) as potentially interesting candidates. For the TREC-7 query 353 (*Antarctica exploration*), GRAS achieves an average precision of 0.5269 as compared to the Porter stemmer, which achieves 0.2866. For this query, the collection contains 114 relevant documents, from which GRAS and Porter stemmer retrieve 111 and 68 documents, respectively. GRAS conflates both *Antarctic and Antarctica* to the same term. In contrast, Porter stemmer puts *Antarctic* and *Antarctica* in two different groups. Such a conflation by GRAS helps retrieve many relevant documents which were not matched when Porter stemmer was used.

In the other direction, consider the TREC-6 query 348 where the performance of Porter stemmer is much better than GRAS, although both the methods retrieve all relevant documents. For this query (Is agoraphobia a widespread disorder or relatively unknown?) the term *agoraphobia* happens to be very important. In this case, GRAS merges *agoraphobia* and *agoraphobic* together, whereas Porter stemmer keeps them separate. No relevant document for this query contains the term *agoraphobic*. Therefore, putting *agoraphobia* and *agoraphobic* in the same class (by GRAS) results in many non-relevant documents at the top of the ranked list, which degrades the overall average precision. Porter stemmer, on the other hand, by adopting a more conservative merging strategy, avoids query-intent drift and consequently populates the top of the ranked list with relevant documents.

Bengali. On Bengali data, GRAS and YASS performed almost equally, with GRAS achieving the highest MAP and having a marginal edge over YASS. Therefore, we focus on some Bengali queries for which the effectiveness of these methods is significantly different. One such example is topic 96 of FIRE 2010 (the longest bar above the x-axis in Figure 13). For this query GRAS achieved a perfect score (MAP) of 1.0, beating YASS which only managed to achieve 0.39. There are only two relevant documents

Table XVIII. Computation Time (in minutes) Taken by Different Methods on Seven Data Sets

Language	# of Words	LING	OARD	YASS	GRAS
Hungarian	528,315	109	1.0	39	2.3
Marathi	854,324	168	2.0	84	1.9
Czech	457,164	81	0.6	27	1.3
English	522,381	95	0.5	41	0.7
Bulgarian	320,673	54	0.7	13	0.9
Bengali	533,605	82	1.4	35	1.1
French	303,349	48	0.4	19	0.4

for this query, and GRAS retrieved them in the first two positions of the ranked list, whereas YASS pulls up many non-relevant documents to the top. The query contains the terms *hatya*, meaning killing, and *hatyakari*, meaning killer, and none of the relevant documents contain the term *hatyakari*, but contain the term *hatya*. GRAS places *hatya* and *hatyakari* in the same class, while YASS puts them in separate classes. The term *hatyakari* is relatively infrequent whereas *hatya* is a frequent term. If they are kept apart (which YASS did), *hatyakari* gets a high term-weight and plays a dominant role in ranking. This phenomenon ultimately caused the promotion of many non-relevant documents to the top, pushing down the relevant documents to lower ranks. For the other queries, we were unable to identify any clear explanation for the performance difference, which seemed to arise out of fairly modest differences in query term weights.

8. EFFICIENCY

In this section, we report and compare the actual processing time taken by the various methods, including the proposed algorithm GRAS. We carried out all our retrieval runs on a computer with a core i3 2.13 GHz processor and 3 GB RAM. Table XVIII clearly shows that GRAS is very fast compared to LING and YASS. Only Oard's method is as efficient as (and sometimes more efficient than) GRAS. Linguistica is very expensive compared to the other methods. The computation time reported for LING and YASS was measured using the implementations provided by the respective authors, whereas we use our own implementation of OARD.

The identification of potential suffix pairs takes about 5 to 15 seconds of computation time for different lexicon sizes (minimum for English, maximum for Hungarian). In GRAS, most of the time is spent on graph construction. Although the Hungarian lexicon is much smaller in size than the Marathi lexicon, the time taken to process the Hungarian lexicon is higher than that for Marathi. Two aspects here influence the processing time: first, the density of the graph, that is, average degree of a node, and second, the length of the suffixes. Hungarian words and suffixes are the longest among the languages studied here and the constructed graph is relatively dense. These are the possible reasons for the relatively slow processing of the Hungarian lexicon.

9. CONCLUSION

We have presented a language-independent statistical stemming algorithm GRAS which is fast as well as effective in the retrieval context. The performance of our algorithm on seven datasets from TREC, CLEF, and FIRE is superior to those of some well-known rule-based and statistical stemming algorithms that have been reported earlier. The proposed method outperforms the rule-based stemmers and a clustering-based statistical stemming algorithm (YASS) on all seven languages. Among these, the performance difference on six languages (Hungarian, Marathi, Czech, English, Bulgarian, and Bengali for rule-based and Hungarian, Marathi, Czech, English, Bulgarian, and French for YASS) was found to be statistically significant. GRAS also performs better than Linguistica, an unsupervised morphological analyzer, on all seven languages. For some languages, the proposed algorithm provides more than 50% performance improvement compared to a baseline strategy that uses unstemmed words. Therefore, our conclusion from the present work is that the proposed stemmer (GRAS) is capable of handling an interesting class of languages and improves performance of mono-lingual information retrieval significantly with a low computation cost.

ACKNOWLEDGMENTS

The authors would like to thank Dipasree Pal for providing the result analysis code.

REFERENCES

AMATI, G. AND VAN RIJSBERGEN, C. J. 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst. 20,* 4, 357–389.

BACCHIN, M., FERRO, N., AND MELUCCI, M. 2005. A probabilistic model for stemmer generation. *Inf. Process. Manage. 41,* 1, 121–137.

DOLAMIC, L. AND SAVOY, J. 2009. Indexing and stemming approaches for the Czech language. *Inf. Process. Manage. 45,* 6, 714–720.

DOLAMIC, L. AND SAVOY, J. 2010. Comparative study of indexing and search strategies for the Hindi, Marathi, and Bengali languages. *ACM Trans. Asian Lang. Inf. Process. 9,* 3.

FRAKES, W. B. AND FOX, C. J. 2003. Strength and similarity of affix removal stemming algorithms. *SIGIR Forum 37*, 26–30.

GOLDSMITH, J. 2001. Unsupervised learning of the morphology of a natural language. *Comput. Linguist. 27,* 2, 153–198.

HARMAN, D. 1991. How effective is suffixing. *J. Amer. Soc. Infor. Sci. 42*, 7–15.

HULL, D. A. 1996. Stemming algorithms—a case study for detailed evaluation. *J. Amer. Soc. Infor. Sci. 47*, 70–84.

KETTUNEN, K. 2009. Reductive and generative approaches to management of morphological variation of keywords in monolingual information retrieval: an overview. *J. Document. 65,* 2, 267–290.

KROVETZ, R. 1993. Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 191–202.

MAJUMDER, P., MITRA, M., AND PAL, D. 2008. Bulgarian, Hungarian and Czech stemming using YASS. In *Proceedings of Advances in Multilingual and Multimodal Information Retrieval: 8th Workshop of the Cross-Language Evaluation Forum*. Springer-Verlag, Berlin, 49–56.

MAJUMDER, P., MITRA, M., PARUI, S. K., KOLE, G., MITRA, P., AND DATTA, K. 2007. YASS: Yet another suffix stripper. *ACM Trans. Inf. Syst. 25,* 4.

MCNAMEE, P. AND MAYFIELD, J. 2004. Character n-gram tokenization for european language text retrieval. *Inf. Retr. 7,* 1-2, 73–97.

OARD, D. W., LEVOW, G.-A., AND CABEZAS, C. I. 2001. Clef experiments at maryland: Statistical stemming and backoff translation. In *Proceedings of the Workshop of Cross-Language Evaluation Forum on Cross-Language Information Retrieval and Evaluation* (Revised Papers). Springer-Verlag, Berlin, U.K., 176–187.

PORTER, M. F. 1997. *An Algorithm for Suffix Stripping*. Morgan Kaufmann, San Francisco, CA, 313–316.

SAVOY, J. 2008. Searching strategies for the Hungarian language. *Inf. Process. Manage. 44,* 1, 310–324.

XU, J. AND CROFT, W. B. 1998. Corpus-based stemming using cooccurrence of word variants. *ACM Trans. Inf. Syst. 16,* 1, 61–81.

Received January 2011; revised July 2011; accepted August 2011