Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Information Processing and Management 41 (2005) 121–137

INFORMATION PROCESSING & MANAGEMENT

www.elsevier.com/locate/infoproman

A probabilistic model for stemmer generation [☆]

Michela Bacchin [*], Nicola Ferro, Massimo Melucci [*]

Department of Information Engineering, University of Padova, via Gradenigo, 6/a, Padova 35131, Italy

Abstract

In this paper we will present a language-independent probabilistic model which can automatically generate stemmers. Stemmers can improve the retrieval effectiveness of information retrieval systems, however the designing and the implementation of stemmers requires a laborious amount of effort due to the fact that documents and queries are often written or spoken in several different languages. The probabilistic model proposed in this paper aims at the development of stemmers used for several languages. The proposed model describes the mutual reinforcement relationship between stems and derivations and then provides a probabilistic interpretation. A series of experiments shows that the stemmers generated by the probabilistic model are as effective as the ones based on linguistic knowledge.
© 2004 Elsevier Ltd. All rights reserved.

Keywords: Multi-lingual information retrieval; Stemming; Mutual reinforcement; Web information retrieval

1. Introduction

An information retrieval (IR) system, which manages text resources, processes words to extract and assign content descriptive index terms to documents or queries. As we use naturally spoken or written language, words are formulated with many morphological variants, even if they are referred to as a common concept. Therefore stemming has to be performed in order to allow words, which are formulated with morphological variants, to group up with each other, indicating a similar meaning. Most of the stemming algorithms reduce word forms to an approximation of a common morphological root, called "stem", so that a relevant document can be retrieved even if the morphology of their own words is different from the one of the words of a given query. It is interesting to note that a stemming algorithm is a special case of query expansion due to the fact that it provides the searchers a way to expand the set of query terms with their morphological variants.

Nevertheless, the effectiveness of stemming is a debated issue, and there are different results and outcomes as reported by Frakes (1992, Chap. 8). If effectiveness is measured by the traditional precision and recall measures, it seems that for a language with a relatively simple morphology, like English, stemming

[☆] This research has stemmed from the work that was presented at the 2002 International Conference on Asian Digital Libraries in Singapore. This paper provides information on new and original ideas in this field of research, in particular it describes a probabilistic model for the methodology which has already been presented in the preceding paper.

[*] Corresponding authors. Fax: +39-049-827-7799.

E-mail addresses: mbacchin@dei.unipd.it (M. Bacchin), nf76@dei.unipd.it (N. Ferro), melo@dei.unipd.it (M. Melucci).

0306-4573/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2004.04.006

has little influence on the overall performance, as reported by Harman (1991). In contrast, stemming can significantly increase the retrieval effectiveness, especially precision, for short queries or for languages with a more complex morphology, like the romance languages, as shown by Krovetz (2000) and Popovic and Willett (1992). Despite this debate, it is commonly accepted that the use of a stemmer is intuitive to many users who can express the query using a specific term without worrying that only a variant of this term can appear in a relevant document (Harman, 1991). Thus, stemming should be a feature of the user interface of an IR service, supplied for example by digital libraries.

To design a stemming algorithm, it is possible to follow a linguistic approach based on prior knowledge of the morphology of the specific language, or a statistical approach which employs some methods based on statistical principles to infer the word formation rules from the corpus of documents. The linguistic approaches are likely to be more effective because the quality of the morphological analysis has been assured by experts in the linguistic field, but the benefits that could be reaped are outweighed by the time necessary to complete the morphological analysis especially when new languages have to be added on an IR service. Furthermore, it is a demanding task to codify all of the word formation rules for languages with a complex morphology and the resulting stemmers can be imprecise; in addition it is not always possible to have an expert for each language. On the other hand, stemming algorithms based on statistical methods ensure no additional costs to add new languages to the system—this is an advantage that becomes crucial, especially for applications like digital libraries which are often constructed for a particular institution or nation, and are able to manage a great amount of non-English documents as well as documents written in many different languages. Of course, the low cost of stemmer generation provided by stemming algorithms based on statistical methods might be counterbalanced by a degradation of the quality of the morphological analysis and, consequently, the retrieval effectiveness.

The research reported in this paper has originated from the study and the experimentation of a graph-based stemming algorithm described by Bacchin, Ferro, and Melucci (2002) and briefly described here. Given a collection of words, we can look at a collection of sub-strings obtained by splitting each word into two parts: the prefix is the first part of the word, and the suffix is the second part. Each prefix or suffix is a node of a graph where each link corresponds to the word obtained by concatenating the linked prefix and suffix. We introduced the notion of mutual reinforcement between stems and derivations, in order to identify the optimal prefixes and suffixes. An optimal prefix corresponds to a stem and an optimal suffix corresponds to a derivation. The rationale to use mutual reinforcement was based on the idea that stems are prefixes which are completed at a high frequency rate by the derivations; derivations, in turn, are suffixes which complete, at a high frequency rate, the stems—in graphical terms, derivations tend to be linked to stems, and vice versa, thus forming communities in the graph. Thus the mutual reinforcement relationship is a kind of *coupled frequent usage* of prefixes and suffixes that are used in order to form words. It is important to note that high frequency of prefixes or suffixes is not a necessary condition nor is it a sufficient condition employed in discovering stems and derivations—very frequently used prefixes are very likely to be stems, but they are discarded if they are not followed by very frequently used suffixes.

The method used to discover the communities of stems and derivations, which are coupled together by mutual reinforcement, is based not only on a computation of the sub-string frequency, as in Hafer and Weiss (1974) or Goldsmith (2001), but also on link analysis algorithms which proved to be effective in Web retrieval. The graph-based stemming algorithm is an instance of affix removal-based stemming and in particular that of suffix stripping—suffix stripping stemming, which splits a word and considers the prefix as the stem, is described in (Frakes & Baeza-Yates, 1992) and adopted by most stemmers currently in use by IR, like those reported by Lovins (1968), Paice (1990), and Porter (1980).

Therefore the main thrust of this paper is to take a step forward from the graph-based stemming algorithm just described and to introduce a probabilistic framework which models the mutual reinforcement between stems and derivations. The idea here is to consider stemming as the inverse of a machine which generates words by concatenating prefixes and suffixes. The paper then shows how the estimation of the probabilities of the model relates to the notion of mutual reinforcement and to the discovery of the communities of stems and derivations. The paper is organized as follows: in Section 2 an intuitive view of the probabilistic framework proposed for stemmer generation is given. The model is formulated in Section 3 and implemented through an algorithm that is described in Section 4; the experiments to assess the performances of the proposed algorithm are described in Section 5; finally Section 6 draws some conclusion and presents the future work.

2. Intuitive view

Many words from some Western languages describe concepts using a root and variations connected to the root. To materialize a concept using one of these languages, a word might be compounded by linking one or more morphemes together. First, a root is probably chosen to select a meaning close to the concept to be expressed, then the chosen root is specialized by picking the derivation which comes closest in fitting the concept to the context. As result, words similar in morphology are often similar in meaning.

Stemming associates each word to a stem which is an approximation of the morphological root that generated a word; it does not aim at finding a linguistically correct root form, but rather a canonical representation for a set of words with similar meaning. According to this view, words can be seen as the outcome of a generative process performed by a hypothetical machine that takes the set of all the possible prefixes and suffixes as input and produces words as output according to some type of linguistic knowledge.

The idea which underlies the intuitive view of the model proposed in this paper is that the machine produces words according to some knowledge about the language, rather than randomly. Thus, a word produced by the machine is the result of joining together a stem and a derivation and not a generic prefix or suffix. This is because the machine is driven by some type of linguistic knowledge. Because of this, the probability of generating a pair is not uniform—since the machine is supposed to correctly operate, the probability that a stem is correctly concatenated with a derivation is higher than the probability that a generic prefix is concatenated with a generic suffix, as shown in Fig. 1.

Stemming can be seen as the inverse of the generative process that has been just described above (Fig. 2). Given a word, a stemmer has to guess the prefix and the suffix in order to form the most probable pair that the machine has chosen to generate the word. As the machine pools together its knowledge of the language, the most probable pair is formed by the stem and the derivation of the word.

3. The probabilistic model

Given a finite collection W of words, let U be the set of N sub-strings generated after splitting each word $z \in W$ into all possible positions, except for those which generate empty sub-strings. From now on, it can be

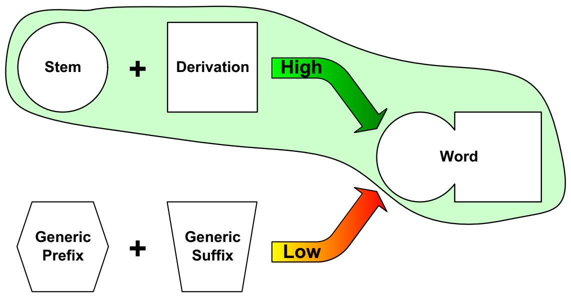

Fig. 1. Representation of the generative process.

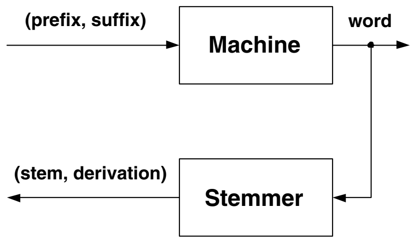

Fig. 2. Relationship between the hypothetical machine and the stemmer.

assumed that words are divided into two non-empty sub-strings only. If x, y are the prefix and the suffix of word z, respectively, then $z = xy$ and there are $n-1$ possible positions to which z is split, if $|z| = n$.

What is known about the machine is that it composes words by concatenating stems and derivations, and that this process is governed by some basic linguistic knowledge. To the observer, the individual pairs of sub-strings composed by the machine are random events. Let us define the universe of the elementary random events as follows:

$$\Omega = \{(x,y) \in U \times U : \exists z \in W, z = xy\}$$

The relation

$$\mathscr{E}(z) = \{(\omega,\omega') \in \Omega \times \Omega : \omega = (x,y), \omega' = (x',y'), xy = x'y' = z\}$$

is defined on the set Ω to express the fact that two elementary events belong to $\mathscr{E}(z)$ if and only if they form the same word z. This is an equivalence relation, indeed the following properties hold:

- reflexive: $\forall\omega \in \Omega$, $xy = xy \iff (\omega,\omega) \in \mathscr{E}(z)$;
- symmetric: $(\omega,\omega') \in \mathscr{E}(z)$ means that $xy = x'y' = z$, i.e., $x'y' = xy = z$; this means that $(\omega',\omega) \in \mathscr{E}(z)$;
- transitive: provided $(\omega,\omega') \in \mathscr{E}(z)$ and $(\omega',\omega'') \in \mathscr{E}(z)$, it follows that $xy = x'y' = z$ and $x'y' = x''y'' = z$ and then $xy = x''y'' = z$, i.e., $(\omega,\omega'') \in \mathscr{E}(z)$.

$\mathscr{E}(z)$ induces a partition of the set Ω because it is an equivalence relation where the sets:

$$\Omega(z) = \{\omega,\omega' \in \Omega : (\omega,\omega') \in \mathscr{E}(z)\}$$

are the equivalence classes of Ω and $\Omega/\mathscr{E}(z)$ is the quotient set. A bijective function $f : W \to \Omega/\mathscr{E}(z)$, which associates each word z to $\Omega(z)$, can be then defined. The set $\Omega(z)$ can be written as:

$$\Omega(z) = \bigcup_{i=1}^{n-1}\{\omega_i\}$$

which is the set of possible concatenations made by the hypothetical machine forming the target word z. Since there is a bijective function between the set of words and the set of the sub-sets $\Omega(z)$,

$$\Pr(z) = \Pr(\Omega(z))$$

where z is the random event that z occurred and $\Omega(z)$ is the event that there exists a pair of sub-strings that generates z. Because the single ω_i are disjoint events and $\Omega(z) = \cup_{i=1}^{n-1}\{\omega_i\}$,

$$\Pr(z) = \Pr(\Omega(z)) = \Pr\left(\bigcup_{i=1}^{n-1}\{\omega_i\}\right) = \sum_{i=1}^{n-1}\Pr(\omega_i)$$

where ω_i is one of the elementary events of $\Omega(z)$.

As stemming can be seen as the inverse of the generative process, a stemmer has to infer the most probable pair of prefixes and suffixes chosen by the machine to generate the given word. Therefore, a stemmer has to compute the expression

$$\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(\omega|z) = \arg\max_{\omega\in\Omega(z)} \Pr(\omega|\Omega(z))$$

to find the "right" split, i.e., the most probable prefix and suffix that generate z. Since ω is an element of $\Omega(z)$ only, $\Pr(\omega|\Omega(z')) = 0$ for any other $z' \neq z$. Therefore, $\Pr(\omega|\Omega(z))$ has to be computed in order to select the most probable split. Using the Bayes' theorem,

$$\Pr(\omega|\Omega(z)) = \frac{\Pr(\Omega(z)|\omega)\Pr(\omega)}{\Pr(\Omega(z))}$$

Note that $\Pr(\Omega(z)|\omega) = 1$, because $\omega \in \Omega(z)$ yields z only, then $\Pr(\omega|\Omega(z)) = \Pr(\omega)/\Pr(\Omega(z))$.

Because the denominator does not influence the ranking of the possible splits of z, the stemmer computes

$$\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(\omega) \tag{1}$$

in order to find the most probable split ω^* of z.

At this point, it is worth noting that the probability of the pair $\omega_i = (x_i, y_i) \in \Omega(z)$ can be expressed in two equivalent ways:

$$\Pr(x_i, y_i) = \Pr(y_i|x_i)\Pr(x_i)$$

and

$$\Pr(x_i, y_i) = \Pr(x_i|y_i)\Pr(y_i)$$

The probability that the hypothetical machine has chosen x_i as prefix and y_i as suffix to generate z, depends on the product of two probabilities—the probability $\Pr(x_i)$ of choosing x_i as prefix and the probability $\Pr(y_i|x_i)$ that y_i is the suffix to complete z. Equivalently, $\Pr(x_i, y_i)$ depends on the probability $\Pr(y_i)$ that y_i is the suffix to complete a word, and the probability $\Pr(x_i|y_i)$ that x_i has been the prefix. Whereas the conditional probability $\Pr(y_i|x_i)$ can be seen as "bridge" to complete z, the probability $\Pr(x_i)$ represents the probability of the first step of word generation. The latter probability is a sort of "pivot" probability, which significantly affects the computation of $\Pr(x_i, y_i)$. Then $\Pr(x_i)$ is the crucial step in the estimation phase. Given the a priori probability $\Pr(y_i|x_i)$, the stemmer looks for $\Pr(x_i)$ that maximizes $\Pr(x_i, y_i)$, i.e., the highest probability that $\omega_i = (x_i, y_i)$ generates z. In other words, the stemmer has to compute

$$\arg\max_{i=1,\dots,n-1} \Pr(y_i|x_i)\Pr(x_i)$$

which is an equivalent expression of $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(\omega)$. The problem is how to estimate these probabilities. Whereas the conditional probabilities can be estimated using the distribution of prefixes and suffixes over the words, the estimation of the probability $\Pr(x_i)$ becomes crucial because it represents the stepping stone of word generation.

At first sight, it would seem that the maximization of $\Pr(x_i)$ is sufficient to maximize $\Pr(x_i, y_i)$ using some maximum likelihood criterion, given $\Pr(y_i|x_i)$. On the contrary, the maximization of $\Pr(x_i)$ is tightly coupled with the maximization of $\Pr(y_i)$, as shown below. To estimate $\Pr(x_i)$ and $\Pr(y_i)$, the following notion of probabilistic mutual reinforcement in stemming is introduced.

> Stems are prefixes which have a high probability of being completed by derivations; derivations, in turn, are suffixes which have a high probability of completing stems.

If a collection of words is observed, a prefix is completed by diverse suffixes, and a suffix completes diverse prefixes. The mutual reinforcement relationship emphasizes that stems are more likely to be completed by derivations; derivations in turn are more likely to complete stems. So if the probability that a prefix is completed by a suffix is high and the probability that the suffix completes the prefix is high, then we can say that the corresponding split is likely to be the right one.

Let us formalize the notion of mutual reinforcement in stemming. It is a fact that $\Pr(x_i) = \sum_{j=1}^{N} \Pr(x_i, y_j)$ and that $\Pr(y_j) = \sum_{i=1}^{N} \Pr(x_i, y_j)$. Note that the $\Pr(x_i, y_i)$s are unknown and so have to be estimated. The fact that $\Pr(x_i, y_j) = \Pr(y_j|x_i)\Pr(x_i)$ and that $\Pr(x_i, y_j) = \Pr(x_i|y_j)\Pr(y_j)$ is further exploited. Thus,

$$\Pr(x_i) = \sum_{j=1}^{N} \Pr(x_i|y_j)\Pr(y_j) \quad i = 1, \ldots, N \tag{2}$$

and

$$\Pr(y_j) = \sum_{i=1}^{N} \Pr(y_j|x_i)\Pr(x_i) \quad j = 1, \ldots, N \tag{3}$$

The mutual reinforcement relationship is given by the fact that $\Pr(x_i)$ is an average mean of the $\Pr(y_j)$s, and at the same time $\Pr(y_j)$ is an average mean of the $\Pr(x_i)$s. Given this relationship, $\Pr(x_i)$ increases as $\Pr(y_j)$ increases, i.e., the higher the probability that x_i is chosen as stem, the higher the probability that its potential suffixes are derivations and that x_i is completed by its potential suffixes. Similarly, the higher the probability that y_i is chosen, the higher the probability that its potential prefixes are stems and that y_i completes its prefixes. Eqs. (2) and (3) highlight that a circular relationship between $\Pr(x_i)$ and $\Pr(y_j)$ exists. To resolve this circularity, an iterative algorithm is proposed in the next Section 4.

4. The algorithm

Given a finite collection of words, for each word the stemmer has to guess, without any intrinsic knowledge of the language, which is the best split for the word, i.e., the most probable pair (prefix, suffix), that can be considered as stem and derivation of the word. This aim can be reached through a two-step algorithm:

- *Global step*: at this step the stemmer considers the whole collection of words and it tries to infer some basic linguistic knowledge from the collection, i.e., the stemmer strives to understand which are the best prefixes and suffixes of U, exploiting equations (2) and (3). Note that this process is independent from the word that the stem has looked for, but it considers the relationships among prefixes and suffixes of the whole collection—this is the reason why this step is called "global".
- *Local step*: at this step the stemmer takes a given word as input and it tries to determine the split which corresponds to a stem and a derivation, using Eq. (1). Although the stemmer uses the linguistic knowledge inferred in the global step, it now operates within a local scope, because it considers only the pairs which lead to the word and not the whole collection.

Thus the stemmer is organized into two parts: the first one performs the global step, while the second one performs the local step, as shown in Fig. 3. In the following, Section 4.1 illustrates the global step, while Section 4.2 explains the local step.

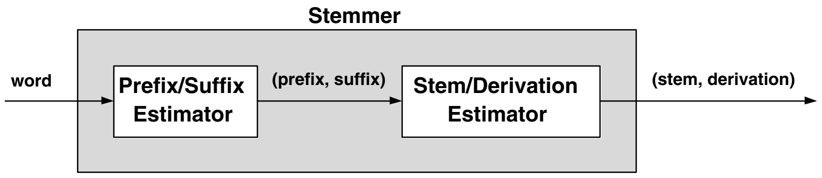

Fig. 3. The organization of the stemmer.

4.1. Global step

To illustrate the algorithm to compute the probabilities, a more compact notation is used than in Eqs. (2) and (3). Let us define

$$\mathbf{p} = [\Pr(x_1) \cdots \Pr(x_N)]'$$

and

$$\mathbf{s} = [\Pr(y_1) \cdots \Pr(y_N)]'$$

i.e., the vector of prefix scores and the vector of suffix scores, respectively. Moreover, let $\mathbf{A} = [a_{sr}]$ be the $N \times N$ matrix such that $a_{sr} = \Pr(x_r|y_s)$, and let $\mathbf{B} = [b_{rs}]$ be the $N \times N$ matrix such that $b_{rs} = \Pr(y_s|x_r)$. Therefore,

$$\mathbf{p} = \mathbf{A}'\mathbf{s}$$

and

$$\mathbf{s} = \mathbf{B}'\mathbf{p}$$

After substituting,

$$\mathbf{p} = \mathbf{A}'\mathbf{B}'\mathbf{p}$$

and

$$\mathbf{s} = \mathbf{B}'\mathbf{A}'\mathbf{s}$$

and then $\mathbf{p}$ is the eigenvector of $\mathbf{C} = (\mathbf{BA})'$ associated to unity eigenvalue, and $\mathbf{s}$ is the eigenvector of $\mathbf{D} = (\mathbf{AB})'$ associated to unity eigenvalue.

An element of $\mathbf{C}$ can be expressed as

$$c_{ij} = \sum_{s=1}^{N} b_{js} a_{si} = \sum_{s=1}^{N} \Pr(y_s|x_j) \Pr(x_i|y_s)$$

which represents the probability that prefix x_i is associated to prefix x_j through their common suffixes. Equivalently, an element of $\mathbf{D}$ can be expressed as

$$d_{ij} = \sum_{r=1}^{N} a_{jr} b_{ri} = \sum_{r=1}^{N} \Pr(x_r|y_j) \Pr(y_i|x_r)$$

which represents the probability that suffix y_i is associated to suffix y_j through their common prefixes.

The algorithm that computes prefix and suffix scores is illustrated in Fig. 4 using pseudo-code.

```
$N$: number of substrings extracted from all the words in $W$
$K$: the number of iterations
$\mathbf{1}$: the vector $(1, ..., 1) \in \mathcal{R}^N$
$\mathbf{s}^{(k)}$: suffix score vector at step $k$
$\mathbf{p}^{(k)}$: prefix score vector at step $k$
$a_{ji} = \Pr(x_i \mid y_j)$
$b_{ij} = \Pr(y_j \mid x_i)$
$\mathbf{p}^{(0)} = \mathbf{1}$
for each $k = 1, ..., K$
    for each $j = 1, ..., N$
        $s_j^{(k)} = \sum_{i=1}^{N} p_i^{(k-1)} \times b_{ij};$
    for each $i = 1, ..., N$
        $p_i^{(k)} = \sum_{j=1}^{N} s_j^{(k)} \times a_{ji};$
    normalize $\mathbf{p}^{(k)}$ and $\mathbf{s}^{(k)}$ so that $1 = \sqrt{\sum_i p_i^{(k)2}} = \sqrt{\sum_j s_j^{(k)2}}$
end.
```

Fig. 4. Compute suffix scores and prefix scores from z.

4.2. Local step

The aim of this step is to choose the most probable split, which corresponds to the prefix that can be the most probable stem of a given word z. In Section 4.1 the aim has been to estimate the marginal probabilities $\Pr(x_i)$ and $\Pr(y_j)$ for all the sub-strings. The conditional probabilities have been used as parameters of the algorithm illustrated in Section 4.1. After computing the marginal probabilities, a prefix probability and a suffix probability are provided for each split of z. If the algorithm implementing the mutual reinforcement relationship had not been employed, the estimation of $\Pr(x_i, y_j)$ would have been computed by using maximum likelihood estimators, which do not embed the mutual reinforcement relationship between stems and derivations. After the algorithm has been performed to disclose and exploit the mutual reinforcement relationship, the marginal probabilities already incorporate the evidence given by the dependencies between prefixes and suffixes. As a result of this, there are different approaches to compute ω^* as from Eq. (1). The following have been identified:

1. $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(x)$, or $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(y)$
   This approach is justified by the fact that the suffix is determined once the prefix x and the word z are given; vice versa for the second equation. This approach selects the best split by selecting the most probable stem, thus disregarding the most probable derivation; vice versa for the second equation.
2. $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(x)\Pr(y)$
   This approach faces the situation from a different point of view: it assumes that, during the estimation of prefix and suffix probabilities illustrated in Section 4.1, the probabilities $\mathbf{p}$ and $\mathbf{s}$ have absorbed some linguistic knowledge, each one on its own, and so x, y can be considered as an independent event and the probability of the generation of the word z is the product of the two marginal probabilities.
3. $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(x)\Pr(y|x)$, or $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(y)\Pr(x|y)$
   The latter two equations view $\Pr(x, y)$ as the combination of one marginal probability and one conditional probability. This approach assumes that the algorithm does not capture the whole mutual reinforcement relationship between x, y and that it is necessary also to consider a stochastic dependence.

In Section 5 the experimental results using two approaches out of the previous three are presented. An exhaustive and in depth analysis of all the criteria is left to future study since it is not the main focus of this paper. The effectiveness of $\arg\max_{\omega\in\Omega(z)} \Pr(x)$ and of $\arg\max_{\omega\in\Omega(z)} \Pr(x)\Pr(y|x)$ are investigated, where $\Pr(x)$ is given by the proposed algorithm and $\Pr(y|x)$ is the reciprocal of the number of words prefixed by x.

5. Experiments

A series of laboratory experiments were conducted according to the Cranfield evaluation model using the procedures and the test data provided by the cross-language evaluation forum (CLEF). CLEF is an initiative for the large scale evaluation of multi-lingual information retrieval systems funded by the European Commission (Peters & Braschler, 2001). The main aim of the experiments was to assess the effectiveness of the stemming algorithm described in the preceding sections by comparing it with stemming algorithms based on linguistic knowledge, i.e., using some basic linguistic rules. The comparison was made on the basis of the effectiveness of retrieval, and specifically on the precision of the retrieval of CLEF documents in comparison to the queries employed during the evaluation campaigns of 2001 and 2002. The variations between the retrieval precision of the system using the linguistic knowledge-based stemmer and the retrieval precision of the system using the algorithm described in Section 4 were observed.

5.1. Statistical methods to validate results

A simple comparison of the variation in percentages between the effectiveness measures of two or more methods does not provide enough information to ensure that the behaviors of the methods are a result of their structural nature and have not occurred by chance. To validate the results in an IR setting, it is necessary to carry out a statistical analysis based on significance testing, as suggested by Hull (1993). To compare two IR methods, A and B, the significance tests were built in a way that the null hypothesis H_0 means that no differences exists between the two methods, and the alternative hypothesis H_1 means that one method is better than the other.

$$\begin{cases} H_0 & : \mu_A = \mu_B \\ H_1 & : \mu_A \neq \mu_B \end{cases}$$

After the formalization of the hypothesis system, a level of significance α has to be chosen, where α represents the maximum probability that one agrees to, associated with the error of rejecting the null hypothesis when in fact it is true. The significance test attempts to disprove the null hypothesis by determining a p-value, which is the probability of an erroneous outcome when the test statistic is calculated on the sample data and suggests that the null hypothesis has to be rejected. In our context, the p-value can be considered as the probability of the observed difference between the performances of IR methods A and B could have occurred by chance. If the p-value is less than α then there is statistical evidence of rejecting the null hypothesis.

The Wilcoxon statistical tests for paired samples were used because the two sets of measures can be considered as measures of different treatments on a set of subjects. This test is a non-parametric statistical test and it did not force us to formulate any assumption on the variable distribution shape.

Carrying on the statistical analysis, two alternative information retrieval methods are compared, considering each query as a statistical unit. Let X_i be the effectiveness measure of the ith query for the first method, Y_i be the same effectiveness measure of the same query for the second method and $D_i = X_i - Y_i$. The *rank* is the consecutive number assigned to a specific observation in a sample of observations sorted by their ascending values, therefore reflecting the ordinal relation of the observation to others in the sample. For each query, this test calculates the ranks of the difference between the two measures. If the rank sum of the positive differences is similar to the rank sum of the negative differences, then the two methods are considered equivalent. The test statistic, whose asymptotic distribution under H_0 is Normal, is

$$T = \frac{\sum R_i}{\sqrt{\sum R_i^2}} \tag{4}$$

where

$$R_i = \text{sign}(D_i) \cdot \text{rank}(|D_i|) \tag{5}$$

When $D_i = 0$ the related statistical units are not considered in the test as reported by Sheskin (1997).

5.2. Method and design of experiments

The aim of the experiments is to evaluate the retrieval effectiveness of the algorithms being proposed, and to compare the level of effectiveness with that of Porter's algorithm, which is based on a priori linguistic knowledge—the algorithm by M. Porter is publicly available at the Snowball Web Site (2003) for research purposes. Our algorithms were compared with the Porter findings because the latter uses a kind of a priori linguistic knowledge of the language, so the comparison with that particular "linguistic" algorithm could give some information about the possibility of estimating linguistic knowledge by statistically inferred knowledge. To test if the system performance was not been significantly hurt by the application of stemming, as hypothesized in Harman (1991), the impact of the stemming algorithms was assessed by comparing their effectiveness with the one reached without any stemmer. The stemming algorithms studied were evaluated by assessing the performances of an information retrieval system in terms of the traditional precision and recall figures. In particular, two measures were used: the average precision (A-P) over the 11 standard recall levels and the R-precision (R-P), which is the precision, averaged over all queries, after R documents have been retrieved, where R is the number of relevant documents for the query, as reported in Voorhees and Harman (2000). The performances of the system were analyzed using standard test collections and changing only the stemming algorithms for different runs, all other things being equal. This way, all the changes in the system performances are just imputable to the stemming process. The evaluation was divided into two stages—at the beginning a global evaluation was carried out taking into consideration the measures averaged over all queries; then, a more analytical study was performed by evaluating the measures query-by-query, i.e., the queries were modelled as statistical units. The Wilcoxon statistical test was applied to check the significance of the results.

5.3. Test data and experimental system

To evaluate the stemming algorithms, as explained above, the performances of an information retrieval system were assessed by using a test collection, changing only the stemming algorithm. The tools used in the experiments are presented in this section.

5.3.1. Test data

The retrieval experiments for the Italian language were conducted using the CLEF 2001 and CLEF 2002 test collection provided by the cross-language evaluation forum (CLEF) consortium, which is reported by Peters and Braschler (2001). In particular the Italian sub-collection of CLEF test collection was used to-

Table 1
Main features of the CLEF collection used in the experiments

	La Stampa	SDA	Total
Size in KB	198,112	87,592	285,704
Number of documents	58,051	50,527	108,578

gether with the 2001 and 2002 query sets. The test documents consist of two distinct sub-sets of articles both referring to 1994:

- *La Stampa*, which is an Italian national newspaper;
- *Italian SDA*, which is the Italian portion of the news-wire articles of SDA (Swiss Press Agency).

The main features of the test collection are reported in Table 1. After a simple case normalization, the Italian sub-collection has a glossary of 333,828 unique words. Both document and query sets are formatted using SGML, and are encoded with the ISO-Latin 1 (ISO-8859-1) character set. Finally, the CLEF collection gives a list of relevance judgments hints which are helpful for the evaluation of the system.

5.3.2. Experimental system

5.3.2.1. IRON. For indexing and retrieval, an experimental information retrieval system, called IRON, was used. IRON was developed by the Information Management Systems (IMS) research group of the Department of Information Engineering of the University of Padova. This software tool is part of the group research work in the area of multi-lingual information retrieval which began in 1999 together with the construction of an Italian test collection for experiments in information retrieval reported by Agosti, Bacchin, and Melucci (1999). IRON is a software tool which has been built on top of the Lucene 1.2 RC4 library which is a high-performance and scalable text search engine library written entirely in Java. Lucene started as an independent project and in September 2001 it became an official Jakarta project. It is a free software application governed by the Apache Software Licence (ASL) and it is publicly available (Lucene Web Site, 2003).

IRON consists of two main modules, Indexer and Searcher, which perform the indexing of the test collection and the retrieval of the information formulated by the queries. The system implements the vector space model, described by Salton and McGill (1983), and a ($\text{tf} \cdot \text{idf}$)-based weighting scheme, reported by Salton and Buckley (1988), as provided by the Lucene library. IRON has been incorporated in our experiments to test the efficiency of the statistical stemming algorithm and it also has been used in the experiments of the IMS group in connection with the CLEF 2002 evaluation campaign, as described by Agosti, Bacchin, Ferro, and Melucci (2003). IRON is written in Java, and hence it can be run over multi-platforms; yet it runs on a machine equipped with a UNIX operating system.

As regards the stop-words used in the experiments, i.e., the words which have little semantic meaning, the Italian stop-list which is available at http://www.unine.ch/info/clef/ was used—it consists of 409 unique words. This stop-list is recommended by the CLEF consortium for the participants of the CLEF campaigns.

5.3.2.2. SPLIT. The algorithm illustrated in Section 4 was implemented using the stemming program for language-independent tasks (SPLIT). From the vocabulary of the Italian CLEF sub-collection, SPLIT spawns a 2,277,297-node and 1,215,326-edge graph, which is processed in order to compute prefix and suffix scores—SPLIT took 2.5 h for 100 iterations on a personal computer equipped with Linux, an 800 MHz Intel CPU and 256 MB RAM.

5.3.2.3. Presentation of the evaluation results. After the indexing of the test collection and the retrieval using each considered stemming algorithm, the standard evaluation software package `trec_eval` 7.0. beta, by Buckley (2003) was used to obtain the set of effectiveness measures to perform the evaluation.

Table 2
Summary of the runs

Label	Description
NoStem	No stemming is used
Porter	Porter's stemming for Italian
S2Maxpref-L3	SPLIT and the $\{\arg\max_{\omega} \Pr(x), \|x\| \geqslant 3\}$ criterion is used to select the best split
S2Maxpref_nlink	SPLIT and the $\{\arg\max_{\omega} \Pr(x)\Pr(y\|x)\}$ criterion is used to select the best split

5.4. Experimental results

Two algorithms based on the methodology being proposed and implemented by the SPLIT software were evaluated; for this reason, the proposed algorithms are often referred to as SPLIT stemming algorithms. This paper presents the evaluation results of two SPLIT stemming algorithms which use different approaches to compute the best split of a word, ω^*:

- $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(x), |x| \geqslant 3$,
- $\omega^* = \arg\max_{\omega\in\Omega(z)} \Pr(x)\Pr(y|x)$.

The former uses a probabilistic approach reported in Section 4 and a heuristic rule which forces the length of the stem to be at least three—in Italian, in fact, there are very few words with a stem shorter than three characters. It was our interested in testing if a simple heuristic could help the performances of the stemming algorithm. For each run, a label and a short description are provided and Table 2 summarizes the labels and the performed runs. The labels have been used through the rest of this section.

A macro-evaluation was carried out by averaging the results of all the queries of the test collection. Out of the 50 queries for 2001 and 2002 query sets, only 47 queries for 2001 and 49 for 2002 had relevant documents in the collection, so only these queries were evaluated in the analysis for that year. Table 3 shows a summary of the figures related to the macro-analysis of the stemming algorithms.

The table provides an overview of the results and suggests formulating the hypothesis that the implemented IR system performed well when using the SPLIT as well as the Porter stemming algorithms. To validate this hypothesis, a more detailed evaluation was conducted query-by-query, i.e., the queries were modelled as statistical units and computed the values of two random variables—precision with SPLIT and precision with the Porter algorithm; in particular the R-P and average-precision figures were computed for each query and for each run. In the following, H_0 is the null hypothesis that the two compared algorithms yield the same level of precision, and H_1 is the opposite of H_0. Then, $\mathscr{A}\{H_0\}$ represents the decision that H_0 is accepted, whereas by $\mathscr{R}\{H_0\}$ is the decision of rejection of H_0. The hypothesis system is as follows:

$$\begin{cases} H_0 & : \mu_{\mathsf{NoStem}} = \mu_{\text{other\_algorithm}} \\ H_1 & : \mu_{\mathsf{NoStem}} \neq \mu_{\text{other\_algorithm}} \end{cases}$$

Table 3
Macro-comparison among runs

	2001			2002		
	Rel. Retr.	A-P	R-P	Rel. Retr.	A-P	R-P
NoStem	1093	0.3387	0.3437	934	0.3528	0.3686
Porter	1169	0.3757	0.3619	972	0.3785	0.3867
S2Maxpref-L3	1151	0.3520	0.3599	930	0.3269	0.3358
S2Maxpref_nlink	1141	0.3684	0.3677	977	0.3682	0.3781

Table 4
A-P: the algorithms compared with the baseline of non-stemming

	Impr.	Equiv.	Decr.	Decision	*p*-Value
2001					
Porter	26	2	19	$\mathscr{A}\{H_0\}$	0.302
S2_Maxpref-L3	25	2	20	$\mathscr{A}\{H_0\}$	0.817
S2_Maxpref_nlink	29	2	16	$\mathscr{A}\{H_0\}$	0.153
2002					
Porter	26	0	23	$\mathscr{A}\{H_0\}$	0.134
S2_Maxpref-L3	21	0	28	$\mathscr{A}\{H_0\}$	0.420
S2_Maxpref_nlink	25	1	23	$\mathscr{A}\{H_0\}$	0.284

Table 5
R-P: the algorithms compared with the baseline of non-stemming

	Impr.	Equiv.	Decr.	Decision	*p*-Value
2001					
Porter	17	15	15	$\mathscr{A}\{H_0\}$	0.513
S2_Maxpref-L3	17	16	14	$\mathscr{A}\{H_0\}$	0.493
S2_Maxpref_nlink	17	29	10	$\mathscr{A}\{H_0\}$	0.171
2002					
Porter	17	18	14	$\mathscr{A}\{H_0\}$	0.322
S2_Maxpref-L3	14	15	20	$\mathscr{A}\{H_0\}$	0.177
S2_Maxpref_nlink	18	19	12	$\mathscr{A}\{H_0\}$	0.422

Table 4 reports the number of queries for which a stemming algorithm could improve, decrease or keep an equivalent average-precision with respect to the non-stemming case, while Table 5 reports the data with respect to the R-P figure. Both tables include the *p*-value. For both A-P and R-P figures the null hypothesis (i.e., the algorithms perform equally) cannot be rejected for all the tested algorithms. Indeed, the *p*-value reported in the last columns of the tables is above the usual α values (0.05 or 0.01) used to decide if the null hypothesis has to be rejected. The number of queries remaining unaltered or which show performance enhancement after the stemming process is greater than the number of queries where precision has decreased, however, the enhancement is not strong enough to be considered statistically significant.

An analysis with respect the Porter algorithm was also conducted. It was our interest in testing if algorithms based on SPLIT could be as effective as one based on prior linguistic knowledge, as the Porter algorithm is. The hypothesis system is as follows:

$$\begin{cases} H_0 & : \mu_{\mathsf{Porter}} = \mu_{\text{other\_algorithm}} \\ H_1 & : \mu_{\mathsf{Porter}} \neq \mu_{\text{other\_algorithm}} \end{cases}$$

Table 6 summarizes the results taking into consideration the number of queries for which the average precision has increased, has remained equal or has decreased when a SPLIT stemmer was used instead of the Porter finding. Table 7 summarizes the results taking the R-precision as measure.

For both effectiveness measures and test collections, the null hypothesis, that Porter and S2_Maxpref_nlink equally perform, cannot be rejected. As regards S2_Maxpref-L3 the Wilcoxon test suggests that H_0 has to be rejected for 2002 and not rejected for the 2001 query set, i.e., S2_Maxpref-L3 performs worse than the Porter algorithm for the 2002 query set, while it performs just as effectively as the Porter algorithm for the 2001 queries.

Table 6
A-P: the algorithms compared with the baseline of the Porter algorithm

	Impr.	Equiv.	Decr.	Decision	p-Value
2001					
S2_Maxpref-L3	16	3	28	$\mathscr{R}\{H_0\}$	0.013
S2_Maxpref_nlink	22	3	22	$\mathscr{A}\{H_0\}$	0.861
2002					
S2_Maxpref-L3	18	0	31	$\mathscr{R}\{H_0\}$	0.019
S2_Maxpref_nlink	23	1	25	$\mathscr{A}\{H_0\}$	0.322

Table 7
R-P: the algorithms compared with the baseline of the Porter algorithm

	Impr.	Equiv.	Decr.	Decision	p-Value
2001					
S2_Maxpref-L3	14	17	16	$\mathscr{A}\{H_0\}$	0.651
S2_Maxpref_nlink	16	18	13	$\mathscr{A}\{H_0\}$	0.387
2002					
S2_Maxpref-L3	9	15	25	$\mathscr{R}\{H_0\}$	0.011
S2_Maxpref_nlink	12	26	11	$\mathscr{A}\{H_0\}$	0.637

In order to carry out a more in depth analysis, the user-oriented point of view was given more emphasis than the system-oriented one. If stemming is applied in an interactive context, as it is applied in digital libraries, the ranking used to display the results to the user takes on more importance. In fact, it would be more an interesting finding that end users can obtain the relevant document after having retrieved 10 or 20 documents instead of after 50% retrieved documents. To assess stemming performance from a more user-oriented point of view, it was in our concern to evaluate how the observed improvement of effectiveness can change the ranking of retrieved documents. The precision values at 10, 20, 30 document cutoff values were observed, as suggested by Harman (1991).

The analysis was carried out first by using the non-stemming case and then by using the Porter case.

The experimental results are summarized with reference to the analysis of cutoff values at different levels, with respect to non-stemming, in Table 8. The symbol "$=$" is used to mean that two algorithms were equivalent, and "$>$" is used to mean that an algorithm was superior to the other. Note that all the statements have been statistically tested for a level of significance $\alpha = 0.05$.

Table 8
Summary of the behavior of the stemming algorithms at different levels of document cutoff value, with respect to the baseline of non-stemming

	P@10	P@20	P@30
2001			
Porter-like	>	>	=
S2_Maxpref-L3	>	=	=
S2_Maxpref_nlink	>	>	=
2002			
Porter-like	=	=	=
S2_Maxpref-L3	=	=	=
S2_Maxpref_nlink	=	=	>

As a further analysis, Tables 9–11 report the number of queries for which the proposed algorithms increase, decrease, or remain unaltered when speaking about the performances of the system with respect to the Porter baseline. A summary of the results for the analysis process performed with respect to the Porter baseline, is reported in Table 12.

Table 9
Precision after 10 relevant documents: the algorithms compared with the baseline of the Porter algorithm

	Impr.	Equiv.	Decr.	Decision	*p*-Value
2001					
S2_Maxpref-L3	11	19	11	$\mathscr{A}\{H_0\}$	0.923
S2_Maxpref_nlink	13	22	13	$\mathscr{A}\{H_0\}$	0.972
2002					
S2_Maxpref-L3	10	19	20	$\mathscr{R}\{H_0\}$	0.022
S2_Maxpref_nlink	15	20	14	$\mathscr{A}\{H_0\}$	0.550

Table 10
Precision after 20 relevant documents: the algorithms compared with the baseline of the Porter algorithm

	Impr.	Equiv.	Decr.	Decision	*p*-Value
2001					
S2_Maxpref-L3	12	16	19	$\mathscr{A}\{H_0\}$	0.102
S2_Maxpref_nlink	13	19	16	$\mathscr{A}\{H_0\}$	0.522
2002					
S2_Maxpref-L3	8	17	12	$\mathscr{R}\{H_0\}$	0.029
S2_Maxpref_nlink	14	21	14	$\mathscr{A}\{H_0\}$	0.895

Table 11
Precision after 30 relevant documents: the algorithms compared with the baseline of the Porter algorithm

	Impr.	Equiv.	Decr.	Decision	*p*-Value
2001					
S2_Maxpref-L3	12	24	11	$\mathscr{R}\{H_0\}$	0.026
S2_Maxpref_nlink	11	25	11	$\mathscr{A}\{H_0\}$	0.711
2002					
S2_Maxpref-L3	10	18	21	$\mathscr{R}\{H_0\}$	0.012
S2_Maxpref_nlink	9	26	14	$\mathscr{A}\{H_0\}$	0.527

Table 12
Summary of the behavior of the stemming algorithms with respect to the baseline of the Porter algorithm

	A-P	R-P	P@10	P@20	P@30
2001					
S2_Maxpref-L3	<	=	=	=	<
S2_Maxpref_nlink	=	=	=	=	=
2002					
S2_Maxpref-L3	<	<	<	<	<
S2_Maxpref_nlink	=	=	=	=	=

Taking into consideration all of the effectiveness measures computed in our analysis, both the stemming algorithms based on SPLIT methodology, S2_Maxpref_L3 and S2_Maxpref_nlink, do not worsen the performances of the system with respect to non-stemming, therefore giving the user a method to expand the query terms with all the word variants, without loss of system performances. If compared with an algorithm based on a priori linguistic knowledge, only S2_Maxpref_nlink, which do not use any heuristic rule, performs as effectively as Porter stemming algorithm, while S2_Maxpref_L3 performs worse.

6. Summary and future work

The probabilistic framework proposed in this paper describes the mutual reinforcement relationship which is the basis for the computation of the probability $\Pr(x)$ that a prefix x is a stem and of the probability $\Pr(y)$ that a suffix y is a derivation. Indeed, the mutual reinforcement relationship can be described by two formulas that inter-relate $\Pr(x)$ and $\Pr(y)$. The experiments confirmed the hypothesis that a stemmer built on the notion of mutual reinforcement relationship is as effective as one built on hand-calculated linguistic rules used for the tested language.

In addition to Italian several other experiments were conducted within CLEF 2003 using other languages, such as English, German, Dutch, Spanish, French. For all the languages tested, the proposed stemmer produced equally good results as those produced by Porter stemmer (Di Nunzio, Ferro, Melucci, & Orio, 2003).

The research work presented in this paper unfolded new problems to resolve and leads to further investigation. The criteria listed in Section 4.2 to implement the local step of the algorithm will be investigated further. It is our intention to consider the problem of decompounding words, which for example is a common phenomenon for German. This issue has been studied also by Braschler and Ripplinger (2003), who demonstrate that the use of decompounding leads to an improvement in retrieval effectiveness.

Acknowledgements

The authors thank Maristella Agosti, Giorgio Di Nunzio, Nicola Orio, and Luca Pretto for the invaluable discussions and suggestions. The authors owe a special thank-you to the reviewers for the useful comments which helped them improve the paper. The research has been partially supported by the Enhanced Content Delivery [1] (ECD) national project.

References

Agosti, M., Bacchin, M., Ferro, N., & Melucci, M. (2003). Improving the automatic retrieval of text documents. In C. Peters, M. Braschler, J. Gonzalo, & M. Kluck (Eds.), *Lecture notes in computer science (LNCS): Vol. 2785. Advances in cross-language information retrieval, third workshop of the cross-language evaluation forum, CLEF 2002, Rome, Italy, September 19–20, 2002. Revised papers* (pp. 279–290). Germany: Springer-Verlag.

Agosti, M., Bacchin, M., & Melucci, M. (1999). Report on the construction of an Italian test collection. In *Joint ACM digital library/SIGIR workshop on multilingual information discovery and AccesS* (*MIDAS*), Berkeley, CA, August 1999. Position paper. Available: http://www.clis2.umd.edu/conferences/midas/papers/agosti.html. Last visit on May 5, 2004.

Bacchin, M., Ferro, N., & Melucci, M. (2002). The effectiveness of a graph-based algorithm for stemming. In E. P. Lim, S. Foo, C. S. G. Khoo, H. Chen, E. A. Fox, S. R. Urs, & C. Thanos (Eds.), *Lecture notes in computer science (LNCS): Vol. 2555. Digital libraries: People, knowledge, and technology. Proceedings of 5th international conference on Asian digital libraries (ICADL 2002), Singapore, December 11–14, 2002* (pp. 117–128). Germany: Springer-Verlag.

[1] http://www-ecd.cnuce.cnr.it/

Braschler, M., & Ripplinger, B. (2003). Stemming and decompounding for German text retrieval. In F. Sebastiani (Ed.), *Lecture notes in computer science (LNCS): Vol. 2633. Proceedings of the European conference on information retrieval research (ECIR), Pisa, Italy, April 14–16, 2003* (pp. 177–192). Germany: Springer-Verlag.

Buckley, C. (2003). The `trec_eval` evaluation package. Available: ftp://ftp.cs.cornell.edu/pub/smart/, 2003. Last visit on May 5, 2004.

Di Nunzio, G., Ferro, N., Melucci, M., & Orio, N. (2003). Experiments to evaluate probabilistic models for automatic stemmer generation and query word translation. In C. Peters, M. Braschler, J. Gonzalo, & M. Kluck (Eds.), *Lecture Notes in Computer Science (LNCS)*, *Evaluation of cross-language information retrieval systems, fourth workshop of the cross-language evaluation forum, CLEF 2003, Trondheim, Norway, August 21–22, 2003. Revised papers*. Germany: Springer-Verlag (in print).

Frakes, W. B. (1992). Stemming algorithms. In W. B. Frakes & R. Baeza-Yates (Eds.), *Information retrieval: Data structures and algorithms*. Englewood Cliffs, NJ: Prentice-Hall.

Frakes, W. B.& Baeza-Yates, R. (Eds.). (1992). *Information retrieval: Data structures and algorithms*. Englewood Cliffs, NJ: Prentice-Hall.

Goldsmith, J. (2001). Unsupervised learning of the morphology of a natural language. *Computational Linguistics, 27*(2), 154–198.

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

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

Hull, D. A. (1993). Using statistical testing in the evaluation of retrieval experiments. In *Proceedings of the ACM international conference on research and development in information retrieval (SIGIR)* (pp. 329–338). Pittsburgh, PA, USA: ACM Press.

Krovetz, R. (2000). Viewing morphology as an inference process. *Artificial Intelligence, 118*, 277–294.

Lovins, J. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics, 11*, 22–31.

Lucene Web Site (2003). Available: http://jakarta.apache.org/lucene/docs/index.html. Last visit on May 5, 2004.

Paice, C. D. (1990). Another stemmer. *ACM SIGIR Forum, 24*(3), 56–61.

Peters, C., & Braschler, M. (2001). Cross-language system evaluation: The CLEF campaigns. *Journal of the American Society for Information Science and Technology, 52*(12), 1067–1072.

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

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–137, Reprinted in K. Sparck Jones, P. Willet, *Readings in information retrieval*, Morgan Kaufmann, 1997.

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

Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. New York, NY: McGraw-Hill.

Sheskin, G. J. (1997). *Handbook of parametric and nonparametric statistical procedures*. Boca Raton, Florida: CRC Press.

Snowball Web Site (2003). Available: http://www.snowball.tartarus.org/. Last visit on May 5, 2004.

Voorhees, E., & Harman, D. (2000). Overview of the sixth text retrieval conference (TREC-6). *Information Processing & Management, 36*(1), 335, Special issue on the sixth text retrieval conference (TREC-6).