A Probabilistic Model for Stemmer Generation

(extended abstract)[⋆]

Michela Bacchin, Nicola Ferro, and Massimo Melucci

Department of Information Engineering – University of Padua
Via Gradenigo, 6/B – 35131 Padova (PD) – Italy
{michela.bacchin, nicola.ferro, massimo.melucci}@unipd.it

Abstract. Today managing textual resources and providing full-text search capabilities on them is a relevant issue also for database management systems. Stemming is part of the indexing and searching processes, when we deal with textual resources. In this paper we present a language-independent probabilistic model which can automatically generate stemmers for several different languages. The variety of word forms makes the match between the end user's words and the document words impossible even if they refer to the same concept - this mismatch degrades retrieval performance. Stemmers can improve the retrieval effectiveness, but the design and the implementation of stemmers requires a laborious amount of effort. The proposed model describes the mutual reinforcement relationship between stems and derivations and then provides a probabilistic interpretation of it. A series of experiments shows that the stemmers generated by the probabilistic model are as effective as the ones based on linguistic knowledge.

1 Introduction

Managing and indexing textual documents is crucial not only for traditional Information Retrieval Systems (IRS) but also for DataBase Management Systems (DBMS), which often provide full-text indexing and search capabilities on the data they manage. Systems like Content Management Systems (CMS) or Electronic Documents Management Systems (EDMS) have to deal with huge amounts of textual data and they are often based on DBMS technology. Many open source and commercial DBMS provide full-text capabilities on the content they manage: for example, PostgreSQL[1] and its module Tsearch2[2], MySQL[3], IBM DB2[4] and its Text Extender[5], Oracle Database[6] and its component Oracle

[⋆] An extended version of this paper has been accepted for publication in *Information Processing & Management*, Elsevier [3].
[1] `http://www.postgresql.org/`
[2] `http://www.sai.msu.su/~megera/postgres/gist/tsearch/V2/`
[3] `http://www.mysql.com`
[4] `http://www.ibm.com/software/data/db2/`
[5] `http://www.ibm.com/software/data/db2/extenders/text/`
[6] `http://otn.oracle.com/products/database`

Text[7], and Microsoft SQL Server[8]. A DBMS, which provides full-text indexing and search functionalities, can enable information access in two distinct and complementary ways: exploiting the structure of the textual data by means of structured queries and exploiting the content of the textual data by means of full-text queries expressed in natural language. Searching documents against natural language queries, arises several issues, but in this paper we will focus the attention on the problems related to the presence, in the text, of several morphological variant word forms which refer to a common concept. The normalization of these word forms to an approximation of the morphological root is called *stemming*, and it aims to allow words to group up with each other, indicating a similar meaning [7]. Basically, a stemming algorithm forms word equivalence classes such that each class includes the words sharing a common *stem*, which is an approximation of the common morphological root. One such class may be, for example, "computer", "computers", "computerized", and "computing" which share the common stem "comput". This way a user can formulate a query without worrying about the morphology of query words, because relevant documents can be retrieved even if the morphology of their own words is different from the morphology of the words of a given query. All of the DBMS mentioned above offer some stemming capability apart from MySQL, which plans to add it in the future.

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 tries to infer the word formation rules from the corpus of documents. The linguistic approaches can be more effective because the morphological analysis is performed 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. 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 a crucial advantage, especially for applications which manage documents written in many different languages.

While stemming is useful in general, there are examples of words for which the decision as to whether stemming should be performed or not is not straightforward. An example is "stocks" which has a meaning in the financial domain if used with "trade", but would have the meaning of "gunstock" in a corpus about weapons. Therefore, the stem "stock" would be good for a domain, but may be bad for another. Statistical algorithms like the one proposed in this paper suit the characteristics of a given text corpus and should produce more effective results [16] than general purpose linguistic algorithms.

[7] `http://otn.oracle.com/products/text/`
[8] `http://www.microsoft.com/sql/`

This paper presents a probabilistic model, which introduces and exploits the notion of mutual reinforcement relationship between stems and derivations, in order to automatically generate stemmers. The notion of mutual reinforcement relationship has been recently investigated in the context of information retrieval. It has been applied to link analysis and hypertext information retrieval by Kleinberg in [10], or has been used for image retrieval purposes by Lempel and Soffer in [11]. In this paper we will investigate if the mutual reinforcement relationship is a suitable notion also for stemming and we will find a positive answer to this question. The paper is organized as follows: Section 2 explains the probabilistic model; the experiments to assess the performances of the proposed algorithm are described in Section 3; finally Section 4 draws some conclusion and presents the future work.

2 The Probabilistic Model

Our stemming algorithm is based on a suffix stripping paradigm, in which each word is split into a pair of substrings, called prefix and suffix, and considers the prefix as the stem. 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 and not at random. 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. Because of this, the probability of generating a pair is not uniform – since the machine exploits some kind of linguistic knowledge, 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. Stemming can be seen as the inverse of this generative process: 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.

Given a finite collection W of words, let U be the set of N substrings generated after splitting each word $z \in W$ into all possible positions, except for those which generate empty substrings. 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 $|z| = n$. Let us define the universe of the elementary random events as follows: $\Omega = \{(x, y) \in U \times U : \exists z \in W, z = xy\}$, and let $\Omega(z) = \{(x, y) \in \Omega : xy = z\}$ be the set of all of the pairs (prefix, suffix) leading to the same word z. The stemmer has to infer the most probable pair of prefixes and suffixes chosen by the machine to generate the given word, computing the expression:

$$\begin{aligned}(x,y)^* &= \arg\max_{(x,y)\in\Omega(z)} \Pr(x,y \mid z) \overset{(a)}{=} \arg\max_{(x,y)\in\Omega(z)} \frac{\Pr(z \mid x,y)\Pr(x,y)}{\Pr(z)} = \\ &\overset{(b)}{=} \arg\max_{(x,y)\in\Omega(z)} \Pr(x,y) \overset{(c)}{=} \arg\max_{i=1,\ldots,n-1} \Pr(x_i,y_i)\end{aligned} \tag{1}$$

where: (a) is obtained applying the Bayes Rule; (b) is obtained observing that $\Pr(z \mid x, y) = 1$, since $(x, y) \in \Omega(z)$ yields to z only, and $\Pr(z)$ is the same for all (x, y) and so it does not influence the maximization; (c) is obtained observing that $\Omega(z) = \cup_{i=1}^{n-1}\{(x_i, y_i)\}$.

In order to estimate the probability distribution of the pairs (x_i, y_i), which is necessary to find the most probable pair, we introduce the following notion of probabilistic mutual reinforcement in stemming:

> 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 $\Pr(x_i, y_j) = \Pr(y_j \mid x_i)\Pr(x_i)$ and that $\Pr(x_i, y_j) = \Pr(x_i \mid y_j)\Pr(y_j)$; furthermore we have that $\Pr(x_i) = \sum_{j=1}^{N}\Pr(x_i, y_j) = \Pr(x_i \mid y_j)\Pr(y_j)$ and $\Pr(y_j) = \sum_{i=1}^{N}\Pr(x_i, y_j) = \Pr(y_j \mid x_i)\Pr(x_i)$. These two latter equations highlight a circular relationship between $\Pr(x_i)$ and $\Pr(y_j)$, which can be resolved by adopting the following iterative approach:

$$\begin{cases} \Pr^{(t)}(x_i) &= \sum_{j=1}^{N}\Pr(x_i \mid y_j)\Pr^{(t-1)}(y_j) \qquad i = 1, \dots, N \\ \Pr^{(t)}(y_j) &= \sum_{i=1}^{N}\Pr(y_j \mid x_i)\Pr^{(t)}(x_i) \qquad j = 1, \dots, N \end{cases} \tag{2}$$

where $t = 1, 2, \dots$ is the iteration index and $\Pr^{(0)}(y)$ is a vector of uniform probabilities and the conditional probabilities $\Pr(x_i \mid y_j)$ and $\Pr(y_j \mid x_i)$ are estimated by the reciprocal of the number of words which end by y_j, and start by x_i respectively. The mutual reinforcement relationship is given by the fact that $\Pr^{(t)}(x_i)$ is an average mean of the $\Pr^{(t-1)}(y_j)$s, and $\Pr^{(t)}(y_j)$ is an average mean of the $\Pr^{(t)}(x_i)$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.

The stemmer is organized as 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 detects the best prefixes and suffixes of U, exploiting equations (2). 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 equation (1). The stemmer uses the linguistic knowledge inferred in the global step, but it now operates within a local scope, because it considers only the pairs which lead to the word and not the whole collection.

3 Experiments

To evaluate the proposed stemming algorithms a series of experiments were conducted according to the Cranfield methodology, based on the usage of *experimental collections* [5, 14]. An experimental collection is a triple $\mathcal{C} = (D, Q, J)$, where D is a set of documents, called also collection of documents; Q is a set of queries, called also topics; J is a set of relevance judgements, i.e. for each topic $q \in Q$ the documents $d \in D$, which are relevant to the query q, are determined. The evaluation of two systems X and Y happens as follows: the document collection D is indexed by both systems X and Y; each system searches the document collection against topics Q and produces, for each topic q, an ordered list of retrieved documents; finally the relevance judgements J allow us to check the two ordered lists of retrieved document and compute the performances of IRS X and Y.

A couple of measures are used for quantifying the performances of an IRS [14]: *recall* which is the proportion of relevant documents which are retrieved; *precision* which is the proportion of retrieved documents that are relevant. Precision and recall are set–based measures, that is they evaluate the quality of an unordered set of retrieved documents. To evaluate ranked lists, precision and recall can be computed for different values of an appropriate parameter: for example, precision can be computed at standard recall levels or after that a given number of documents has been retrieved, which is called precision at different *document cut–off values* (DCV).

For the experiments reported in the following, we used the test data provided by the Cross-Language Evaluation Forum[9] (CLEF) [12], and specifically the Italian CLEF collection for the 2001 and 2002 evaluation campaigns [1, 2].

Experimental Settings We measured the performances of an IRS where only the stemming algorithm has been changed 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 analysis was carried out computing the average precision (A-P) over the 11 standard recall levels ($0\%, 10\%, 20\%, \ldots, 100\%$ of relevant documents) and the precision computed at 10, 20, 30 DCV (P@10, P@20, P@30) [15]. The last figures allow us to carry out a more user-oriented analysis: indeed it can be an interesting finding that end users can obtain relevant documents after having retrieved 10 or 20 documents instead of after 50%

[9] `http://www.clef-campaign.org/`

	2001 A-P	2002 A-P
NoStem	0.3387	0.3528
Porter	0.3757	0.3785
SPLIT	0.3684	0.3682

Table 1. Global comparison among runs.

	2001				2002			
	A-P	P@10	P@20	P@30	A-P	P@10	P@20	P@30
Porter vs NoStem	=	>	>	=	=	=	=	=
SPLIT vs NoStem	=	>	>	=	=	=	=	>
SPLIT vs Porter	=	=	=	=	=	=	=	=

Table 2. Statistical analysis among runs.

retrieved documents [8]. All the effectiveness measures were computed using the standard evaluation software package `trec_eval`[10].

We tested two algorithms: our algorithm based on the probabilistic model, called *Stemming Program for Language Independent Tasks* (SPLIT), and Porter's stemming algorithm [13], which is a widely used stemming algorithm based on linguistic knowledge; we used the Italian version of Porter's stemmer[11].

Experimental Results A global evaluation was carried out by averaging the results of all the queries of the test collection. The results, which are reported in Table 1, suggested the hypothesis that the IR system performed well when using SPLIT as well as Porter's stemming algorithms. To validate this hypothesis, the queries were modelled as statistical units and the Wilcoxon test was applied to check the statistical significance of the results [9]. The null hypothesis H_0 means that two compared algorithms yield the same level of precision, and H_1 is the opposite of H_0. Table 2 reports the results of this analysis where the symbol "=" means that two algorithms were equivalent, and ">" ("<") means that an algorithm was superior (inferior) to the other; all the statements have been statistically tested for a level of significance $\alpha = 0.05$. For the Average-Precision figure we cannot reject the null hypothesis, that the two stemming algorithms performed equally to no-stemming; on the other hand, for both P@10 and P@20 figures of the 2001 runs both SPLIT and Porter's stemming algorithms performed better than no-stemming. This is an important result because the use of stemming makes the system more user-friendly and it can be applied without loss of performances or even with an improvement in some cases. Furthermore, recalling the importance of the P@10 and P@20 figures from an user-oriented standpoint, the two stemming algorithms were able to improve the performances

[10] `ftp://ftp.cs.cornell.edu/pub/smart/`

[11] `http://www.snowball.tartarus.org/`

for the highly ranked documents, which are scanned first by an end-user. Table 2 compares also the SPLIT algorithm with the Porter's algorithm: for all the effectiveness measures and test collections, we cannot reject the null hypothesis, that Porter's and SPLIT algorithms equally perform. Thus we found that our statistical algorithm can be as effective as the Porter's one.

Summing up, the SPLIT algorithm does not worsen or even enhances the performances of the system with respect to no-stemming and thus it gives the user an intuitive method to expand the query terms with all the word variants. If compared with an algorithm based on a-priori linguistic knowledge, it performs as effectively as Porter's stemming algorithm.

4 Summary and Future Work

The probabilistic model proposed in this paper describes the mutual reinforcement relationship between stems and derivations, which is the basis for automatically generate stemmers. The experiments confirmed the hypothesis that a stemmer built on the notion of mutual reinforcement relationship is as effective as one built on hand-coded linguistic rules used for the tested language. In addition to Italian several other experiments were conducted within CLEF 2003 using other languages [6], such as English, German, Dutch, Spanish, French. For all the languages tested, the proposed stemmer produced equally good results as those produced by Porter's stemmer.

The research work presented in this paper unfolded new problems to resolve and leads to further investigation. For example, the notion of mutual reinforcement relationship could be generalized from two components, i.e. a stem and a derivation, to n components. This way the probabilistic model could be applied to the decompounding problem, that is splitting word-compounds made by two morphemes or more – very frequent for germanic languages – in order to improve retrieval effectiveness [4].

Acknowledgments

The authors thank Maristella Agosti, Giorgio Di Nunzio, Nicola Orio, and Luca Pretto for the invaluable discussions and suggestions. The research has been partially supported by the *Enhanced Content Delivery*[12] (ECD) national project. The authors wish to thank the anonymous reviewers for their useful comments, which will be taken into account also for future developments of the work.

References

[1] M. Agosti, M. Bacchin, N. Ferro, and M. Melucci. Improving the Automatic Retrieval of Text Documents. In C. Peters, M. Braschler, J.Gonzalo, M. Kluck (eds.) *Advances in Cross-Language Information Retrieval, Third Workshop of the Cross-Language Evaluation Forum, CLEF 2002. Rome, Italy, September 19-20, 2002. Revised Papers*, pages 279–290. Lecture Notes in Computer Science (LNCS) 2785, Springer-Verlag, Germany, 2003.

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

[2] M. Bacchin, N. Ferro, and M. Melucci. 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, and C. Thanos (eds.) *Digital Libraries: People, Knowledge, and Technology. Proceedings of 5th International Conference on Asian Digital Libraries (ICADL 2002), Singapore, December 11-14, 2002*, pages 117–128. Lecture Notes in Computer Science (LNCS) 2555, Springer-Verlag, Germany, 2002.

[3] M. Bacchin, N. Ferro, and M. Melucci. A Probabilistic Model for Stemmer Generation. *Information Processing & Management*, Elsevier (in print).

[4] M. Braschler and B. Ripplinger. Stemming and Decompounding for German Text Retrieval. In F. Sebastiani (ed.) *Proceedings of the European Conference on Information Retrieval Research (ECIR), Pisa, Italy, April 14-16, 2003*, pages 177–192. Lecture Notes in Computer Science (LNCS) 2633, Springer-Verlag, Germany, 2003.

[5] C. W. Cleverdon. The Cranfield Tests on Index Languages Devices. In K. Spack Jones and P. Willett, editors, *Readings in Information Retrieval*, pages 47–60. Morgan Kaufmann Publisher, Inc., San Francisco, California, USA, 1997.

[6] G. Di Nunzio, N. Ferro, M. Melucci, and N. Orio. Experiments to Evaluate Probabilistic Models for Automatic Stemmer Generation and Query Word Translation. In C. Peters, M. Braschler, J. Gonzalo, and M. Kluck (eds.) *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.* Lecture Notes in Computer Science (LNCS), Springer-Verlag, Germany, (in print).

[7] W.B. Frakes. *Stemming algorithms.* In Information Retrieval: Data Structures and Algorithms. W.B. Frakes and R. Baeza-Yates, Eds. Prentice Hall, Englewood Cliffs, NJ, chapter 8, 1992.

[8] D. Harman. How Effective is Suffixing? *Journal of the American Society for Information Science*, 42(1):7–15, Wiley, 1991.

[9] D.A. Hull. Using Statistical Testing in the Evaluation of Retrieval Experiments. In *Proceedings of the ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 329–338, Pittsburgh, PA, USA, ACM Press, 1993.

[10] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, September 1999.

[11] R. Lempel and A. Soffer. PicASHOW: Pictorial Authority Search by Hyperlinks On the Web, *ACM Transactions on Information Systems*, 20(1):1–24, ACM Press, January 2002.

[12] C. Peters and M. Braschler. Cross-Language System Evaluation: the CLEF Campaigns. *Journal of the American Society for Information Science and Technology*, 52(12):1067–1072, Wiley, 2001.

[13] M.F. Porter. An Algorithm for Suffix Stripping. *Program*, 14(3):130–137, 1980. Reprinted in K. Sparck Jones, and P. Willet, *Readings in Information Retrieval*, Morgan Kaufmann, 1997,

[14] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, USA, 1983.

[15] E. Voorhees and D. Harman. Overview of the Sixth Text Retrieval Conference (TREC-6). *Information Processing & Management*, 36(1):335, Elsevier, 2000. Special Issue on the Sixth Text Retrieval Conference (TREC-6).

[16] J. Xu and W.B. Croft. Corpus-based stemming using cooccurrence of word variants, *ACM Transactions on Information Systems*, 16(1):61–81, ACM Press, January 1998.