Detecting Inflection Patterns in Natural Language by Minimization of Morphological Model *

Alexander Gelbukh,[1,2] Mikhail Alexandrov,[1] SangYong Han [2+]

[1] National Polytechnic Institute, Mexico
www.Gelbukh.com, dyner1950@mail.ru

[2] Chung-Ang University, Korea
hansy@cau.ac.kr

Abstract. One of the most important steps in text processing and information retrieval is stemming—reducing of words to stems expressing their base meaning, e.g., *bake*, *baked*, *bakes*, *baking* → *bak*-. We suggest an unsupervised method of recognition such inflection patterns automatically, with no a priori information on the given language, basing exclusively on a list of words extracted from a large text. For a given word list V we construct two sets of strings: stems S and endings E, such that each word from V is a concatenation of a stem from S and ending from E. To select an optimal model, we minimize the total number of elements in S and E. Though such a simplistic model does not reflect many phenomena of real natural language morphology, it shows surprisingly promising results on different European languages. In addition to practical value, we believe that this can also shed light on the nature of human language.

1 Introduction

Nowadays huge amounts of information are available in more and more languages. For example, in May 2004 the number of official languages of the European Union reached 20 and will grow soon. The need of processing multidisciplinary documents in so many languages results in growing interest to knowledge-poor methods of text processing.

One of the most important modules in a system dealing with natural language, such as information retrieval or document classification system, is stemming. The task of a stemmer algorithm is to map the words having the same base meaning but differing in grammatical forms, to the same letter string that can be used to identify the word independently of its morphological form. As the common identifier of a set of word-forms, their common initial substring is usually used, e.g., *bake*, *baked*, *bakes*, *baking*

* Work done under partial support of the ITRI of Chung-Ang University, Korea, and for the first author, Korean Government (KIPA) and Mexican Government (SNI, CONACyT, CGPI-IPN). The first author is currently on Sabbatical leave at Chung-Ang University.

+ Corresponding author.

A. Gelbukh, M. Alexandrov, S.Y. Han. Detecting Inflection Patterns in Natural Language by Minimization of Morphological Model. In: Progress in Pattern Recognition, Image Analysis and Applications (CIARP 2004). Lecture Notes in Computer Science, N 3287, Springer-Verlag, 2004, p. 432–438.

→ *bak*-. Often morphological derivations of the word are included in the set: *bake*, *baked*, *bakes*, *baking*, *baker*, *bakery* → *bak*-.

Though the quite problem is important for English, it is much more important for processing texts in very many other languages, among which are almost all European languages. Indeed, while in English there are only four morphological variants of a verb, in Spanish verbs have 65 forms, while in Russian 250 (mostly due to participles), which are to be mapped to a common stem by the stemmer.

Manual construction of the corresponding dictionaries or rules is a tedious and labor-consuming task, especially for languages for which little linguistic resources are available (there are ca. 5,000 languages in the world). An attractive alternative is automatic learning of the necessary models from the texts themselves.

The languages spoken in the world can be roughly classified as follows:

- Inflective languages. Words in such languages consist of a stem and a number of suffixes and/or prefixes. The number of suffixes (or prefixes) for words of a given part of speech is fixed (or several different variants may exist), thus the number of different combinations of suffixes (prefixes) is fixed, e.g.: Eng. *ask-ed*, Span. *pregunt-aba-s* 'you were asking', *pregunt-e-n-se* 'please you (many) ask yourself'. Most European languages, except for Finnish, Hungarian, and Basque, are of this type.
- Agglutinative languages. Words in such languages consist of a stem and a potentially infinite number of suffixes attached to it as needed, e.g. Turkish *Türk-yali-lastir-a-ma-di-k-lar-i-mi-z-dan* '*one of those that we could not have possibly turned into a Turkish*', with the stem *Türk*- and a set of suffixes. Examples of such languages are Hungarian, Turkish, Korean, Aztec, etc.
- Isolating languages. Words in such languages do not change, so that each word is its own stem. Examples of such languages are Chinese or Vietnamese.
- Intraflective languages. In such languages the root meaning of a word is expressed with consonants, while the grammatical variations with vowels intermixed with the consonants, e.g., Arabic *kitab* 'book' consists of a stem *k-t-b* 'book-' and a morpheme -i-a- expressing grammatical meaning. Examples of such languages are Arabic and Hebrew.
- Incorporating languages. In such languages words consist of many stems glued together by complicated rules. Such a word represents the meaning of a whole sentence. Examples of such languages are Sanskrit, Chukchee, and some North American native languages.

In this paper we only deal with inflective languages, though we believe that our methods can be adjusted to the languages of other classes. We present an unsupervised approach to automatic construction of a stemmer basing only on a list of words extracted from large enough text.

The paper is organized as follows. In Section 2 we discuss the previous work on the topic, in Section 3 we explain our algorithm, and in Section 4 present the experimental results. Section 5 draws conclusions and lists some future work directions.

2 Previous Work

There are three main approaches to stemming:

- Dictionary-based,
- Rule-based, and
- Statistical-based.

Dictionary-based approach It provides the highest quality of results at the cost of the highest development expenses. This approach implies the development of a dictionary listing all known words of a given language along with their inflection classes and other necessary information for generation of all their morphological forms [2]. A theoretical advantage of a dictionary-based approach is that it deals correctly with the words that look like inflected forms but in fact are not. For example, *darling* looks like a form of a verb **to darl*, while in fact it is not.

However, the main advantage of the dictionary-based approach is its correct treatment of exceptions, which can be individual (*men* → *man*) or regular (*stopping* → *stop*, Span. *conozco* 'I know' → *conocer* 'to know', Rus. *molotka* 'of hammer' → *molotok* 'hammer').

A weakness of the dictionary-based approach is the treatment of words absent in the dictionary. In this case one usually has to resort to a supplementary rule-based algorithm. Obviously, the need to develop, maintain, and process in runtime a large dictionary and/or a complex analysis system is the main practical disadvantage of such approaches.

Rule-based approach This approach can be well exemplified by the well-known Porter stemmer [4]. This stemmer uses a complex cascades system of manually tuned rules such as:

1. (*v*) *ING* →
2. *BL* → *BLE*
3. (**d* and not (**L* or **S* or **Z*)) → single letter
4. (*m* = 1 and **o*) → *E*

etc.

The left-hand side of these transformation rules is a condition that fires the rule. It is an expression, usually containing a pattern to be matched with the string at hand. If the condition is met, the corresponding part of the string is substituted with another substring. The first rule above describes deletion (empty right-hand part) of a suffix *-eng* given that the string contains a vowel (v) preceding this suffix, possibly separated from it by an arbitrary substring (*). The second rule prescribes addition of -e after *-bl*. The third rule deletes repetition of the consonant in the words like *stopped*; *d* stands for this double consonant. In the fourth rule, *m* stands, roughly speaking, for the number of non-ending vowels, and *o* for a special form of the last syllable of the string.

The rule-based approach is much less expensive in terms of necessary linguistic resources, and yet powerful enough to correctly process many of regular exceptions;

other exceptions can be treated with a small dictionary. Still, rule-based approach requires detailed analysis of the linguistic properties of the language at hand and careful manual construction of the rules.

Statistical-based approach This approach allows for fast and totally automatic development of a stemmer for a new language. Most approaches of this type use supervised learning techniques, which rely on a set of manually prepared training examples [1]. However, collecting and selection of such training examples can be problematic. In addition, the absence of examples of a specific type can lead to lacunas in the learned data resulting in massive errors.

In this paper we suggest an unsupervised approach to learning stemming rules from a list of words extracted from a corpus of the given language. Since the approach is unsupervised, it does not rely on subjective expert judgments. Just because of this, we believe that the possibility of learning morphological information from the texts without human intervention can shed some light on the nature of human languages.

As an example of a previous work on unsupervised learning of morphology [3] can be mentioned. However, unlike [3], we do not apply complex heuristics and do not use statistical considerations. Instead, we try to find the absolute minimum number of the elements (stems and endings, not letters) which describe the given language.

3 The Algorithm

Given a word list, we find the set of possible stems and endings of the language at hand. Then, we can decompose any word—either from the same list or an unseen one—into a stem and ending. In case of ambiguity we select a combination of the most frequent stem and ending (the global ambiguity can be solved by mutual reinforcement method). Below we concentrate on the problem of finding the sets of stems and endings of the given language.

Problem formulation We rely on the following two hypotheses:

1. The words of the language are simple concatenations of one stem and one (possibly complex) suffix (or prefix). Thus, we currently ignore any sandhi phenomena (such as *lady* + *es* = *ladies*), suppletivism (*foot* / *feet*) and other complications of real language morphology.
2. Language is constructed in such a way that minimal learning effort is necessary for its acquisition; in particular, it has the minimal necessary amount of stems and endings. The stems and endings are "re-used" to form many combinations: *ask*, *ask-ed*, *ask-ing*, *bak-e*, *bak-ed*, *bak-ing*.

Mathematically, the task of finding the corresponding set of stems and endings can be formulated as follows: Given a set V of letter strings extracted from a text, find two sets of strings, S (standing for stems) and E (standing for endings), such that any word $w \in V$ is a concatenation of a suitable stem and ending: $w = s + e$, $s \in S$, $e \in E$, and

$|S| + |E|$ has the minimum value over all sets with such properties, where $|X|$ is the number of elements in the set X. In other words, find minimum sets S and E generating V, i.e., such that $V \subseteq S + E$.

If we suppose in addition that the language is suffixal (and not prefixal), we can additionally require that of all possible pairs S and E with the same $|S| + |E|$ preferable are those with smaller $|E|$.

Genetic algorithm Unfortunately, we are not aware of a less-than-exponential algorithm for finding the sets S and E. So we conducted experiments using a genetic algorithm to find an approximate solution; any other method of optimization could be used as well.

First, we experimented with chromosomes of the length $|V|$ whose genes are the points of division of the individual wordforms from V; for example, a gene 3 at the position corresponding to *darling* stands for the division hypothesis *dar-ling*. We used simple crossover and random mutation. For each such set of division hypotheses, we calculated the total number of stems $|S|$ and endings $|E|$. To reflect our preference for smaller E (versus smaller S), we considered as fitness function

$$|S| + 0.000001|E| \rightarrow \min,$$

the coefficient only affecting the choice between chromosomes with the same $|S| + |E|$.

However, such a search space proved to be too large. To reduce the search space for sake of performance, we considered chromosomes with binary values indicating the presence or absence of a certain stem or ending in S or E. For this, we construct the maximal sets S' and E' of all possible prefixes and all possible suffixes of all strings from V. From them, we remove all those elements that occur only once. Indeed, a decomposition of a $w \in V$ into $w = s + e$, where either s or e occurs only once, can be substituted by a decomposition $w = w + \lambda$, where λ is an empty ending, without changing $|S| + |E|$.

Note that when an element is excluded from S' or E', the frequency of another element—the second half of the decomposition of a word w—decreases and can become 1, so that this element will also be excluded. Such iterative exclusion of elements from S' and E' further reduces their size, finally producing the sets S'' or E'' with such a property that for any element $s \in S''$ there are at least two different $e \in E''$ such that $s + e \in V$, and similarly for E''.

With this, we form binary chromosomes of the length $|S''| + |E''|$ so that a value of 1 stands for the inclusion of the corresponding element in S or E, correspondingly. If for a given selection of S and E, some word $w \in V$ cannot be decomposed, we consider the whole word w a new element of S. This gives us the following fitness function:

$$|S| + 0.000001|E| + |V \setminus (S + E)| \rightarrow \min.$$

The last member of the expression stands for the non-decomposable (with the given random selection of S and E) words from V, which we add to S on the fly.

Since the search space with such a method is considerably reduced, we can find better approximate solutions. In addition, we observed significant gain in quality when we removed not only the endings occurring once but all too rare endings, e,g.,

all endings occurring less than |V|/1000 times. This is possible since we suppose that all endings used in the language are rather productive (are used many times).

Note that due to eliminating the stem or ending candidates that occur only once, our algorithm will usually correctly deal with difficult words such as *darling*. Indeed, even if *-ing* is a likely ending, *darl-* is not a frequent stem. Thus, the algorithm will prefer decomposition of this word into the stem *darling-* and an empty ending.

4 Experimental Results

We applied our algorithm, as described at the end of the previous section, to the official list of words permitted in crossword games such as Srabble. The list has 113,809 wordforms; the found |S| + |E| = 60917. Here is an example of the divisions obtained:

abject-	*abjur-ing*	*ablaze-*	*abluent-s*	*abnegat-ing*
abject-ly	*abla-te*	*ab-le*	*ablush-*	*abnegat-ion*
abjectness-	*abla-ted*	*ablegate-*	*ablut-ed*	*abnegat-ions*
abjectness-es	*abla-tes*	*ablegate-s*	*ablut-ion*	*abnormal-*
abjurat-ion	*abla-ting*	*able-r*	*ablut-ions*	*abnormalit-ies*
abjurat-ions	*abla-tion*	*able-s*	*ab-ly*	*abnormalit-y*
abjur-e	*ablation-s*	*able-st*	*abmho-*	*abnormal-ly*
abjur-ed	*ablativ-e*	*ablings-*	*abmho-s*	*abnormal-s*
abjur-er	*ablativ-es*	*ablins-*	*abnegat-e*	
abjur-ers	*ablau-t*	*abloom-*	*abnegat-ed*	
abjur-es	*ablaut-s*	*abluen-t*	*abnegat-es*	

One can see that the results are not perfect but quite promising. Note that the examples shown here is not an optimal solution; if we run our genetic algorithm for a longer time, we will find a better solution. As compared with Porter stemmer [4] the result is not as good. However, unlike manually tuned Porter stemmer, our algorithm was presented with only 114 thousand of English wordforms and found the presented decomposition in a fully unsupervised manner.

We also apply the same algorithm to a small Spanish wordlist extracted from *Don Quijote*, of only 22,966 words; the found |S| + |E| = 7959. Here is an example of obtained decomposition:

ablándate-	*abolengo-*	*aborrec-en*	*ab-ra*	*abra-semos*
abland-áis	*aboll-é*	*aborrec-ió*	*abracé-*	*abrazándo-le*
abland-ó	*abolla-da*	*aborrec-ible*	*ab-ran*	*abrazándo-nos*
abland-aba	*abolla-do*	*aborrec-ida*	*abra-só*	*abrazándo-se*
abland-aban	*abomin-ábamos*	*aborrec-ido*	*abra-sa*	*abrazáro-nle*
abland-ado	*abomin-able*	*aborrec-idos*	*abrasa-da*	*abrazáro-nse*
abland-an	*abomin-ado*	*aborrec-imiento*	*abrasa-das*	*abrazár-selos*
abland-ar	*abomin-o*	*aborrezco-*	*abrasa-dores*	*abra-zó*
abland-ara	*abon-asen*	*abr-áis*	*abra-sados*	*abrazól-a*
abland-arme	*abon-o*	*abr-i*	*abrasa-n*	*abrazól-e*
abland-aron	*aborrascadas-*	*ab-ría*	*abra-sar*	*abrazól-os*
abland-arte	*aborrec-í*	*ab-rían*	*abrasa-rla*	*abraza-ba*
abland-e	*aborrec-ía*	*abrí-la*	*abrasa-rnos*	*abra-zada*
abobado-	*aborrec-e*	*abr-ís*	*abrasa-sen*	*abra-zado*

abraza-miento	*abraza-rá*	*abraza-rme*	*abraza-se*
abraza-ndo	*abraza-ra*	*abraza-ron*	*abr-azo*
abra-zar	*abraza-rle*	*abraza-rse*	*abr-azos*

One can observe that accent alternations in Spanish verb stems present certain problems to our algorithm. However, given so small word list and the fact that the presented solution is not optimal (which can explain some random anomalies), the results seem promising.

We also apply our algorithm to some other inflective languages, such as Russian, with similar results.

5 Conclusions and Future Work

We have presented an unsupervised algorithm for recognizing the morphological structure of an inflective language, with application to stemming. Currently our algorithm ignores many phenomena of the real natural language morphology, such as sandhi (including Spanish accent alternations), suppletivism, or letter-phoneme correspondence; dealing with such phenomena will be a topic of our future work. We also believe that a similar approach can be applied to some other types of languages, such as agglutinative ones, with suitable modifications.

An interesting direction of future work is to detect, in an unsupervised manner, the syntactic classes of words, roughly corresponding to parts of speech. This can be done by clustering the contexts where the words with certain endings occur. With this, we expect to improve the behavior of our model on difficult words such as *darling*.

We will also try different algorithms for finding better approximate solutions problem formulated in Section 3, for example, simulated annealing.

References

1. M. Alexandrov, X. Blanco, A. Gelbukh, P. Makagonov. Knowledge-poor Approach to Constructing Word Frequency Lists, with Examples from Romance Languages. *Procesamiento de Lenguaje Natural* **33**, 2004.
2. A. Gelbukh, G. Sidorov. Approach to construction of automatic morphological analysis systems for inflective languages with little effort. In: Computational Linguistics and Intelligent Text Processing (CICLing-2003). *Lecture Notes in Computer Science* **2588**, Springer-Verlag, pp. 215–220.
3. J. Goldsmith. Unsupervised Learning of the Morphology of a Natural Language. *Computational Linguistics*, **27** (2), 2001.
4. M.F. Porter. An algorithm for suffix stripping. *Program*, **14** (3): 130–137, 1980.