# Automatic Learning of Stemming Rules for the Indonesian Language

**Lily Suryana Indradjaja**
School of Computing
National University of Singapore
3 Science Drive 2, Singapore 117543
g0303015@nus.edu.sg

**Stéphane Bressan**
School of Computing
National University of Singapore
3 Science Drive 2, Singapore 117543
steph@nus.edu.sg

## Abstract

We present a method for the automatic learning of stemming rules for the Indonesian language. The learning process uses an unlabelled corpus. In the first phase the candidate (word, stem) pairs are automatically extracted from a set of online documents. This phase uses a dictionary but is nevertheless not trivial because of morphing. In the second phase the rules are induced from the thus obtained list of pairs of words with their respective stems. We evaluate the effectiveness of our method for different sizes of the training set with different settings on the thresholds of the support and confidence of each rule. We discuss how these variables affect the quantity and quality of the rules produced.

## 1 Introduction

The Internet and other modern communication infrastructures can be the media for either the predominance of cultural globalization or for the blooming of cultural diversity and culture interchange. The choice of the kind of information society that we are creating is, in part, in the hands of technologists and researchers devising techniques and designing tools for facilitating the maintenance, management of multi-cultural information. One of the central issues in the management of multicultural information or the management of information in a multicultural context is the processing of natural language. There are still more than 6000 living languages at the dawn of the twenty first century. Tools for the processing of speech and text often require large collections of reference data such as dictionaries, morphological rules, grammars, and phonemes. Such collections are expensive to build since they normally require human expert intervention.

Our research is concerned with the economical and therefore semi-automatic or automatic acquisition of such linguistic information necessary for the development of other-than-English or multilingual information systems. Our motivation comes from the desire to build computational linguistics and information retrieval tools for the processing and retrieval of documents written in the Indonesian language. Indonesian is the official language of the republic of Indonesia. Although several hundreds regional languages and dialects are used in the Republic, it is spoken and understood by more than two hundred and forty million people. Indonesian is written using roman script and there are more than 1,000,000,000 documents in Indonesian Language on the public World Wide Web.

Stemming is the process of finding the stem (root) of a word, by stripping away the affix attached to the word. In many languages words are often obtained by affixing existing words or roots. Stemming is particularly useful in applications in which morphologically-related words are processed or treated in the same way regardless of their forms, for example in a text classifier, information retrieval system, and in dictionary lookup tool. Studies have shown that in application like information retrieval system, inflectional morphology plays a more important role as compared to derivational morphology. This is because derivational morphology gives rise to the generation of words belonging to different classes, which sometimes lead to a big difference in meaning. For example, consider the phrase "compute matrix product" and "usage of matrices in computational linguistic". When we use these two phrases as a search string in some search engine, we would expect *matrix* and *matrices* to be regarded as the same word. However, if we consider computational and compute as the same word, a search on "compute matrix product" will generate pages related to computational linguistics, which are quite unrelated to what we actually search for.

However, in a dictionary lookup tool, particularly one for the Indonesian language, it turns out that both types of morphology are significant. The words listed as entries in an Indonesian dictionary are usually root words; the various affixes applicable to a root word are usually listed together in the corresponding entry. Thus, to find a word in the dictionary, we must first find its stem before proceeding to find the corresponding entry of the root word in the dictionary.

Most stemming rules in the Indonesian language are simple. Prefixes and suffixes are attached to the front and the back of a word. However, there are certain characteristics of the stemming rules that complicate the process. Plurals are usually presented as duplications of the stem. Infixes, inherited from the Javanese language, although not frequently used anymore, are also known. Affixing frequently results in the transformation of a letter of the root word. The transformation can be in the form of morphing, insertion, or deletion of letter(s) as follows:

- **Morphing of letter(s)**
  Morphing in the Indonesian language usually applies to the first letter of the root word, in which that letter is replaced by some other letter(s).
  An example of morphing is when the prefix "me-" is attached to the word "tari", the letter 't' in the stem is morphed into 'n' such that the affixed word would be "menari". Another example, "me-" attached to "sapu" becomes "menyapu" (the letter 's' is replaced by 'ny').
- **Insertion of letter(s)**
  When the prefix "me-" is attached to the word "baca", the letter 'm' is inserted in between the prefix and the stem, such that the affixed word would be "membaca".
- **Deletion of letter(s)**
  When a prefix ending with a certain letter is attached to a stem starting with the same letter, one of the duplicate letters is discarded. For example, the prefix "ber-" attached to "renang" becomes "berenang" instead of "berrenang".

| Stem's First Letter | Morphing | Insertion |
|---|---|---|
| a,i,u,e,o,g,h,q | -- | 'ng' |
| b,f,v | -- | 'm' |
| c,d,j,z | -- | 'n' |
| k | 'ng' | -- |
| l,m,n,r,w,y | -- | -- |
| p | 'm' | -- |
| s | 'ny' | -- |
| t | 'n' | -- |

***Table 1:** Morphing and Insertion Rules as a Function of the First Letter of a Stem.*
Note: There is no morphing/insertion rule for root words starting with the letter 'x', as no such word is known in the Indonesian language.

Keraf (1999) shows that all of the known suffixes and most of the prefixes in the Indonesian language can be attached to a word without undergoing any modification, except for the following prefixes: (1) the prefixes "me-" and "pe-" are those that require morphing and insertion of letter(s) to the affixed word, (2) "ber-" and "ter-" require deletion of the first letter of the stem when attached to a word starting with the letter 'r'. It turns out that we can define a fixed set of morphing and insertion rules according to the first letter of the stem (see Table 1).

There have been some research efforts carried out on stemming Indonesian words; most of them work by manually defining a set of stemming rules and using them to find the possible stem(s) of a word. In this paper, we try to automatically induce the stemming rules from a corpus. The reason why we are interested in learning the rules automatically is because the set of affixes in the language is not fixed; the language itself and its usage evolve and grow. Indonesian is also the host of several sublanguages (frequently dubbed as *bahasa gaul*) particularly popular among younger generations.

These forms are frequently used on the Internet in chat rooms, newsgroups, websites and emails. Their informality defeats the conventional language tools. Our approach aims at being adaptive to these changes. For example, we observe that several words ending with the letters "ai" become words ending with "e" substituted to the original "ai". Examples of such words are pakai → pake, satai → sate, sampai → sampe. The ability to find such rules would enable us to develop tools extending our ability to process Indonesian documents from those written in a formal style to those, and they may well form the majority, written in casual style. The next section of this paper describes some research efforts related to this subject. Section 3 presents our proposed approach: the extraction of the candidate [word, stem] pairs and the learning of the stemming rules from the set of candidates. Section 4 reports some results demonstrating the effectiveness of the method. We conclude by summarizing our results and pointing at current and future work.

## 2 Related Work

### 2.1 Morphology Learning

Gaussier (1999) presents an unsupervised method to learn suffixes and suffixation operations from an inflectional lexicon of a language. The goal is to build the derivational families of a language. The intuition behind the extraction of suffixes is that long words of a given language tend to be obtained through derivation, and more precisely through suffixation, and thus could be used to identify regular suffixes. Acquisition of suffixes automatically as well as inducing a linguistically motivated structure in a lexicon are the primary novelties of this work.

Schone and Jurafsky (2000) present a work for morphology knowledge-free induction. They introduce a semantic-based algorithm for learning morphology, which only proposes affixes when the stem and the affixed word are sufficiently similar semantically. The algorithm focuses on inflectional languages. The authors claim that combining their semantic approach with frequency based approaches play complementary roles. They show that their semantics-only approach provides morphology induction results that rival systems such as Goldsmith (2000) and Gaussier (1999). Results are shown only for English.

Goldsmith (2001) describes unsupervised algorithms for extracting morphological rules from a corpus having no prior knowledge of the language, using Minimum Description Length (MDL) analysis (Rissanen, 1989). The input is a corpus of words in the language to be studied, and the desired output is an analysis as close as possible to the analysis made by a human morphologist. The goal of this approach is to find "stems" and categorize words into classes, while determining the morphemes which define each class. There is no intention to handle morphophonemic alternations, and thus the learnt morphology is limited to prefixes and suffixes.

The research efforts presented above are similar to what we have in mind. However, they are mostly carried out on English or other European language. They do not address the problem of transformations, specifically morphing, existing in the Indonesian language.

Yona and Wintner (2002) propose the development of a system which can learn (semi-automatically) the morphological rules of the Hebrew language, expressing the rules in a manner which is easy to maintain, using the supervision of a native speaker who is not necessarily a programmer or a linguist. However, it does not address the problem of obtaining a large enough corpus that contains enough examples for the learning algorithms to be able to recognize the processes in Hebrew inflectional morphology.

### 2.2 Indonesian Stemmers

There have been several implementations of Indonesian stemmer. There is one devised by a student of University of Indonesia (Siregar, 1995), and another developed by Kent Ridge Digital Lab, Singapore. There is also an Indonesian dictionary built by the Indonesian Agency for the Assessment and Application of Technology (http://nlp.aia.bppt.go.id/kebi/), which incorporates a stemmer when looking up a word. All of the stemmers mentioned above are dictionary-based stemmers, i.e. the stemming process is assisted by a dictionary to check whether the candidate stem is a valid Indonesian word.

Vinsensius (2001) implements a non-dictionary-based stemmer, which works roughly like the Porter stemmer for English. This stemmer is developed for an information retrieval system for the Indonesian language.

The choice of whether to use a dictionary in the stemming process depends on the purpose and needs of the application that will be using the stemmer. In a stemmer built for an information retrieval system, a root word found by the stemmer might be invalid (i.e. not found in the dictionary). But it does not matter as in such system we are concerned more with finding the presence of morphological relations among words. For example, the Porter stemming algorithm will stem the words "dies" and "died" to "di". Although "di" is not a valid English word, the fact that "dies" and "died" are actually stemmed to the same word is enough for the information retrieval system.

The stemmer that we will build here is intended to serve as a basis for tools like dictionary lookup and spelling checker, in which the root words found by the stemmer need to be valid words. Studies have shown that for such purpose, a stemmer needs to be assisted by a dictionary in order to produce high accuracy. This fact is also verified by the work in (Ahmad et al, 1996) which showed that high error rates and incomprehensible stems are obtained when no dictionary is used in stemming Malay words. Since the Indonesian and Malay language have a lot in common, we believe that this claim is also true for the Indonesian language. As such, in building our stemmer, we use a dictionary obtained from http://www.seasite.niu.edu/Indonesian/.

## 3 Proposed Approach

### 3.1 Stemming Rules

The stemming rules we wish to learn are of the form: *affix* / **{Z=(*root_condition*).X}** *A (.A)** ;

Dot (.) indicates string concatenation. *affix* is the symbolic representation of the type of affix applicable in a rule. **Z** is the root word. *root_condition* is the set of letter(s) that a root word must begin with in order for the rule to be applicable. The presence of this set is optional. **X** is the substring of the root word, which is obtained by stripping away the first letter(s) in the root_condition. *A* is a non-terminal symbol which could be one of the following: a prefix/infix/suffix, or '-' (for duplication), **Z**, or **X**. Star (*) indicates 0 or more occurrences of the set in the bracket.

The following are example of such rules:

**me- / {Z=(t).X}   men.X** ; This rule is interpreted as follow: when the prefix "me-" is attached to a stem starting with the letter 't', the first letter of the stem is morphed into 'n' such that the affixed word would be "men-" concatenated with the rest of the stem. Example: "me" + tari   menari.

**me- / {Z=(c|d|j).X}   men.Z** ; When the prefix "me-" is attached to a stem starting with the letter 'c', 'd', or 'j', the letter 'n' is inserted between the prefix and the stem in the affixed word. Example: "me-" + cuci   mencuci.

### 3.2 Extracting Candidate Stem

We automatically extract the candidate [word, stem] pairs by finding all the possible substrings of a word that match some words in the dictionary. Note that we employ a dictionary (containing only a list of Indonesian root words) in this process to simplify the problem. We can actually compare a word with another word in the corpus (instead of comparing it with a word in the dictionary), but it would take a lot of time to find two words that have some substring in common, and it would also generate a lot more [word, stem] pairs that are possibly false (i.e. the word and the stem are actually morphologically unrelated), and this would further complicate the process of learning the rules.

### 3.3 Learning Rules from the Set of Candidates

We give our learner the set of terms with their candidate stems. The learner starts by trying to find the similarity between a word and its stem letter by letter, and generate a candidate rule to consider. For each candidate, the learner runs through the list of the words in the training set to see how many examples a rule can cover. Since we need to handle transformations depending on the first letter of the root word, the learner also records the first letters of the stems of each covered examples and stores this information in some set. If it later turns out that the size of this set is too big, the learner will conclude that a rule might not need any condition to be imposed on the root word in order for the rule to be applicable to the word.

For each rule, we record its support and confidence, defined as follows: Support = | E+| / | N | and Confidence = | E+|/| E | where E is the set of examples in the corpus having the same symbolic representation in a rule and satisfying the conditions on the root word, if any; E+ is a subset of E, the set of examples which is positively covered by a rule (i.e. the application of the rule to the root in the example will generate the same affixed word); and N is the set of examples used in training. We set a threshold on the value of support and confidence that a rule must pass in order to be considered valid.

## 4 Experiments and Results

### 4.1 Corpus

The corpus that the system uses to learn the stemming rules consists of a list of words together with their respective stems and affixes. The words are extracted from the online news and documents obtained from http://www.mediaindo.co.id. There are altogether 9898 documents used here.

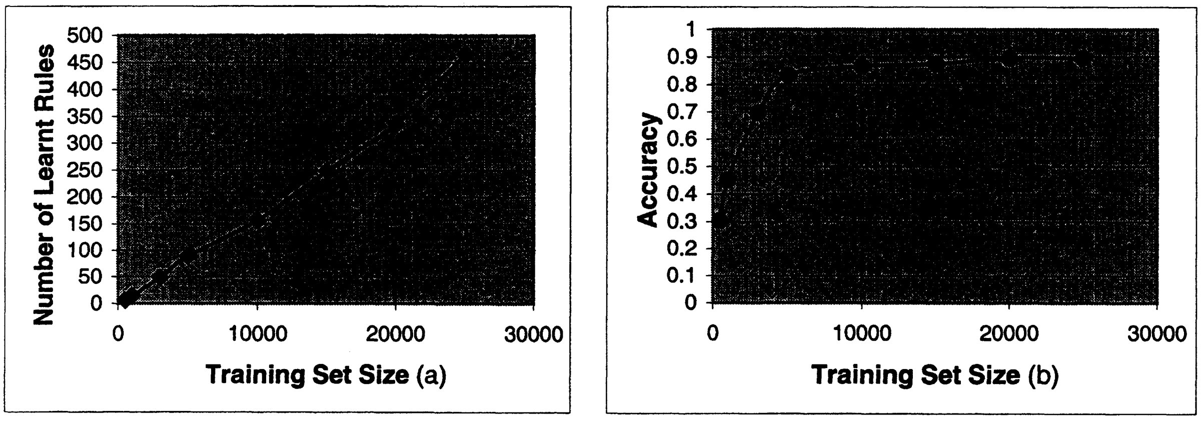

*Figure 1: The effect of varying the size of training set on (a) the number of rules produced, and (b) the accuracy of the learnt rules (accuracy). Support threshold value is set at 0.0001.*

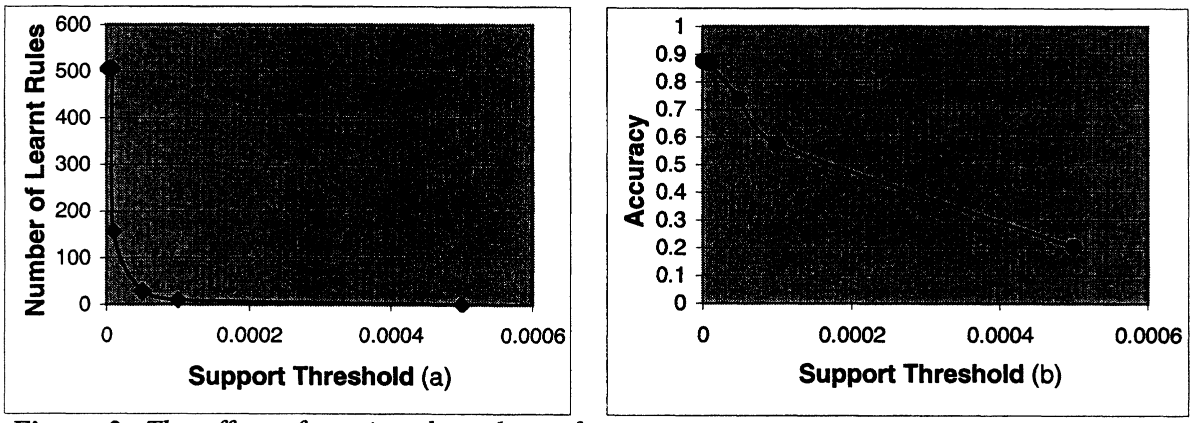

*Figure 2: The effect of varying the values of support threshold on (a) the number of rules produced, and (b) the accuracy of the learnt rules (accuracy). Training set size is 10,000.*

## 4.2 Candidate Stems

From a list of 67,403 words, we obtain 236,805 candidate [word, stem] pairs. Comparing this number with the number of [word, stem] pairs obtained by an existing stemmer (subsequently called the standard set of [word, stem] pairs), which is 10 to 20 times less, shows that we have actually generated a lot of new [word, stem] pairs. Some correspond to new affixes (see conclusion for the example of the new affixes) but many are not examples of useful [word, stem] pairs. This situation is inevitable since we have to handle morphing and thus we have to consider every possible replacement of the first letter(s) of the stem. The subsequent learning algorithm is able to filter out the false examples with proper support and confidence thresholds.

## 4.3 Stemming Rules

Using the inductive algorithm described in the previous section, we generate the stemming rules. Preliminary experiments show that the confidence threshold does not have significant effect on the result. We therefore only use two parameters, i.e. the training set size and the support threshold. The learnt rules are then tested on the candidate [word, stem] pairs. Given a test case, we take the affixed word and get the possible set of stems using an existing set of rules (subsequently called the original set of rules) and the set of learnt rules. Several metrics can be used to evaluate the performance of the algorithm. In this paper we define the accuracy as the completeness of the application of the learnt rules with respect to the original set of rules, i.e. the percentage of words in the corpus in which the stems generated by the learnt rules match those generated by the original set of rules. Figures 1 and 2 report the number of rules learned and their accuracy with respect to the size of the training set and the confidence threshold, respectively. From these figures we see that the learning algorithm can achieve up to 90% accuracy. Manual inspection of the learnt rules also shows that there are some new affixes which are not included in the original set of rules, but are discovered by the algorithm. Some examples are the suffixes –isme, –wan, –wati.

## 5 Conclusion

We have presented an approach to the automatic extraction of stemming rules from a corpus. The approach can achieve up to 90% accuracy, even with a small corpus of less than 10,000 words. The method is adaptive and can learn new forms, spelling mistakes, usages, colloquial forms. Although the corpus we have used is in standard Indonesian, our method has discovered some affixes rarely documented and which were not included in the original set of rules. Although we do not report the detailed results in this paper, we have tried our method with other languages with different morphology such as Italian and Filipino. 80 to 90% of the rules obtained match known stemming/affixation rules for those languages.

Affixes in Indonesian are essentially derivational. They represent the various parts-of-speech played by words in the same family. We conjecture that the part-of-speech of the root word might be significantly useful in determining the part-of-speech of the affixed word, and similarly, the correct stem of a word with several candidate stems may be determined by looking at the part-of-speech of each of the candidate stems. We are therefore currently studying part-of-speech and part-of-speech tagging in correlation with stemming in the Indonesian language.

Details and further reading of the results and findings in this paper can be found in Indradjaja (2003).

### References

Ahmad, F., Yusoff, M., and Sembok, T. M. T. 1996. Experiments with a Stemming Algorithm for Malay Words. *Journal of the American Society for Information Science, Vol 47(12):909-918.*

Gaussier, Éric. 1999. Unsupervised Learning of Derivational Morphology from Inflectional Lexicons. *ACL '99 Workshop Proceedings: Unsupervised Learning in Natural Language Processing.* University of Maryland.

Goldsmith, John. 2000. Linguistica: An automatic morphological analyzer. *Proceedings of 36th meeting of the Chicago Linguistic Society.* Chicago.

Goldsmith, John. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics, 27(2):153–198, June.*

Indradjaja, Lily Suryana. 2003. Computational Linguistics for the Indonesian Language. *Honours Year Thesis, National University of Singapore.* Singapore.

Keraf, Gorys. 1999. Tata Bahasa Rujukan Bahasa Indonesia. *PT Gramedia Widiasarana Indonesia.* Jakarta, Indonesia.

Schone, Patrick and Daniel Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. *Proceedings of CoNLL-2000 and LLL-2000, pages 67–72.* Lisbon, Portugal.

Siregar, Neil Edwin F. 1995. Pencari Kata Berimbuhan pada Kamus Besar Bahasa Indonesia dengan Menggunakan Algoritma Stemming. *Final Year Thesis, University of Indonesia.* Jakarta, Indonesia.

Vinsensius Berlian Vega S. N. 2001. Information Retrieval for the Indonesian Language. *M.Sc. Thesis, National University of Singapore.* Singapore.

Yona, Shlomo and Wintner, Shuly. 2002. Supervised Learning of Morphological Rules for Hebrew. *Graduate Thesis Proposal, Haifa University.* Israel.