Learning Morphology: Algorithms for the Identification of the Stem Changes

Evelin Kuusik
Institute of the Estonian Language
Roosikrantsi 6, Tallinn EE0100, Estonia
e-mail: evelin@kki.ee

The aim of the current work is to create tools for the automatic recognition of the Estonian stem changing rules. The main problem consists in bringing together the formal classification features available to the computer and classification based on human knowledge. This paper introduces two algorithms. First, in STLearn the supervised inductive learning technique is used to find out the suitable features for automatic recognising of the stem changes. Two stem variants can be bounded by more than one stem change. The second algorithm is created for the identifying the whole set of rules for stem pairs.

Current work is a part of a project based on the open model of language [Viks94] according to which all regular and productive phenomena of the natural language are represented by different types of rules and irregular phenomena are listed in small dictionaries - exception lists. This approach gives opportunity to process the regular words not listed in dictionaries - new derivatives, loan-words etc.

Subsystem of morphology plays the central role in processing of the morphologically complex languages as the Estonian language is.

The number of possible stem variants can strongly vary in Estonian: in some inflection types there are no stem variants at all, in some of them a word can have even five different regular stem variants. Current work presents tools for creating a formal description of the Estonian stem changing rules, starting from the pair of the stem variants. The Concise Morphological Dictionary of the Estonian (CMD) [Viks92] serves as a bases for current work and contains over 36 000 headwords, each of them has two stem variants on the averages.

The principle types of changes are the following:

1. Stem-grade changes. Stem can occur either in a strong or a weak grade; the grades are differentiated first of all by phonetic quantity (2nd or 3rd degree of quantity marked by') that may be accompanied by various sound changes enfolding the medial sounds. For instance members of the stem pair *hõive-h'õive* are distinguished only by the different phonetic quantity; in case of couple *aabe-'aape* the rewriting rule $b \rightarrow p$ is concurrent with the phonetic quantity change.
2. Stem-end changes. Stem can appear either as a lemmatic stem or an inflection stem; stem variants are differentiated by changes enfolding the final sounds (e.g. *'aadel-aadli, j'alg* \foot\-*j'alga, sipelgas* \ant\-*sipelga*).
3. Secondary changes. These changes are conditioned by the certain context arising after either the stem-end or the stem-grade change (e.g. *k'uppel* \dome\ $\rightarrow$ **k'uppli* $\rightarrow$ *k'upli*).

About 20 % of stems stay changeless, mostly take place the stem-end or stem-grade changes or both at the same time.

Formally the recognition of the stem change rules can be reduced to the classification task with string pairs as the objects to classify and possible rules of stem changes as the classes. System has to create class descriptions from the 'available' data: characters and their belongness to the sound classes. The important demand to the classification system is the linguistical

correctness [Kuus95]. Because of that the technique of inductive supervised learning is the most suitable for the current task.

Inductive supervised learning (learning from examples) is one of the main techniques in machine learning. Given a set of examples and counterexamples of a class the learning system induces a general class description that covers all of the positive examples and none of the counterexamples.

Most of the learning algorithms assume the attribute-value pairs as input, with fixed number of attributes and known set of values for every attribute. In case of data, presented as strings, class is often defined by the substring varying in length. Determining is the main attribute (in our case, character corresponding to the changing sound) and its direct environs - context that consists of the complement attributes. In most cases the width of the determining context is unknown at first - the learning system has to deal with undefined number of the attributes.

The main specifying operation in case of string data is the adding an attribute - extending the context. As the length of the strings can be very different and in most cases strings are relatively long, then the learning direction towards expanding the context is preferable. The other way, to consider all string as context at first and try to specify the class description by dropping the redundant attributes are much more complex (complexity depends directly on string length).

The algorithm is designed to find class description for each of the seven formal rules of stem changes:

1. deletion of the single character;
2. insertion of the single character;
3. replacement of the single character by the single character;
4. replacement of the two characters by the single character;
5. replacement of the single character by the two characters;
6. replacement of the two characters by the two characters;
7. replacement of the (final) suffixes.

The six rules correspond to the stem-grade changes, the last one corresponds to the stem-end changes. The seven formal rules of stem changes described above make up the set of classes. As at the same time only one of these can be recognised is suitable to join the single class descriptions into decision list covering the whole set of examples. Decision list is the sequence of the *if then... else...* clauses arranged according to the generality level of the conditions, while the last condition (class description for the stem-end changes in current case) is the constant true. In other words if between two stem variants no stem-grade changes is observed, the stem-end change holds between them.

For each stem-grade changing rule the system has to create the description differentiating stem variant pairs placed under it from all others. Main attributes are the character(s) corresponding to changing sound in the first string and the character(s) in the same position(s) in the second one. Therefore the description is represented as a disjunction of the conjunctions:
$((a_i \in \text{consonant}) \wedge (b_i \in \text{vowel})) \vee ((a_i \in \text{vowel}) \wedge (b_i \in \text{vowel}))$ ...

Inductive learning system needs the domain expert who gives the possible classes and provides each class with examples - objects belonging to this class. Usually the counterexamples - objects which do not belong to the class are given too. In this work positive examples are the stem variant pairs subordinating to current stem changing rule. Counterexamples are positive examples of all remaining classes. The initial description hypothesis takes into account only the characters correponding to the changing sound. The description is specified by adding conditions - extending the context.

Context can be extended in two directions: left, to the beginning of the stem and right, towards the end of the stem. Preferred is the direction in case of which the cover extent of the discriminative

conjunctions are higher. If the cover extents are equal, the domain specific heuristics is used according to which the left context (context enfolding the medial sounds) is more informative and the left extension is choosen.

The algorithm
Notation:
P^+ - set of conjunctions valid for the positive examples only
P^- - set of conjunctions valid for the negative examples only
C - composing description
C' - current conjunction

STLearn
Init (P^+, P^-)
$C=P^+\backslash P^-$
While $P^+ \cap P^- \neq \varnothing$
Get_left_context (P^+, P^-)
Get_right_context (P^+, P^-)
Select_successor (P^+, P^-, C')
$C=C' \cup C$
Update (N^+, N^- P^+, P^-)
EndWhile
Optimize (C)

Procedure Init sets up the initial class descriptions taking into account only the characters corresponding to the changing sound.

Procedure Update refreshes the sets of examples. Discriminative conjunctions are added to the class description and examples corresponding to them are removed from the training set

Procedures Get_Right_Context and Get_Left_Context extend the conjunctions by adding correspondingly the right and left context.

Procedure Select_successor selects the best of two extended conjunctions.

Procedure Optimize generalises final class description to the terms of the sound classes.

Morphological and phonological phenomena are usually described by rewriting rules. To adhere to this tradition in current work the stem changing rules are represented in the following way:
$R : a \rightarrow b \,/\, C_l \_ C_r$
This says that the string a is to be replaced by (rewritten as) the string b whenever it is preceded by C_l (the left context) and followed by C_r (the right context). If a is equal to the empty string, then rewriting operation is reduced to the insertion, the same for the string b means deletion. As it is mentioned the stem variants can be bounded by more than one stem change. For instance between stem variant pair *sepp* \smith\ - *sepa* hold two rules:
stem-end change 0 → a / _#
stem-grade change pp→p / e_a

Therefore the decision list compiled by STLearn should be used in cycle until all stem changes are identified. From the viewpoint of the modelling of the natural language stem changes system the rule sets holding for stem variant pairs are more informative than single rules. Each established rule is immediately applied on the first string and the algorithm continues with intermediate word form (that may not exist in real language) until the first string becomes equal to the second one. Domain theory says that in Estonian can only one stem-end and/or stem-grade change appear between the pair of the stem variants. As the string pair is parsed from left to right the stem- grade change is observed before the stem-end change. In the case of the rule set containing the both type of rules the contexts need some correction (because some secondary changes in medial sounds can take place only after applying stem-end changes). To correct the contexts the rules are once more applied in right order (stem-end change at first) and the contexts are updated.

Generating the rule set:
Let $R=\varnothing$ be the initial rule set, a and b are the stem variants and r is the current rule.
1. Search the stem-grade or stem-end change;

2. Form the corresponding rule r;

2.1. add rule to the rule set R;

2.2. apply the rule r to the string a;

3. If $a=b$ then stop, otherwise search the secondary changes;

4. If secondary change is observed

4.1. add the corresponding rule to the set R;

4.2. apply the rule r to the string a;

5. If $a=b$ then stop, otherwise go to the step 1;

6. If $|R|>1$ then update the contexts.

The test results of the algorithms described in the current work show that it is possible to classify the stem changes according to formal features available from the text and at the same time to do it correctly in linguistic sense. Class descriptions were formed using the 540 training exmples. Recognition algorithm was tested on 56120 stem variant pairs of CMD, linguistically incorrectly classified pairs were not observed.

Further work provides designing the methods for the generalisation of acquired rule sets into formal grammar in term of the sound classes and elicitation of the corresponding exception lists. This description will be adequate to the open model of language.

References

[Kuus95] E.Kuusik. Automatic identification of the Estonian stem variation rules. In *Short Papers Presented at the 10th Nordic Conference of Computational Linguistics Nodalida-95.* Helsinki 1995, 86-89.

[Viks92] Ü.Viks. A Concise Morphological Dictionary of the Estonian. Tallinn, 1992.

[Viks94] Ü.Viks. A morphological analyzer for the Estonian language: the possibilities and impossibilities of automatic analysis. In *Automatic Morphology of Estonian 1.* Tallinn,1994.