Using the Lemmatization Technique for Phonetic Transcription in Text-to-Speech System

Abstract. This paper deals with lemmatization technique and its using for the phonetic transcription of exceptional words. The lemmatizer is based on language morphology and uses the lexicon of word basic forms and inversion of the derivation rules to acquire the lemmatization rules which are essential for finding the word bases. We have described the lemmatization algorithm and necessary modifications of the lemmatizer to transcribe exceptional words. The main goal of the designed system is memory saving of the exceptional lexicon. The experimental results have shown that we can save from 18.3% (English) to 98.4% (Finnish) of size of the full lexicon. Hence, this system is suitable for high inflectional and agglutinative languages.

1 Introduction

Phonetic transcription is an important issue for systems which deal with the spoken language. Especially, it is used to convert a written text into a sequence of phonetic symbols in the text-to-speech systems (TTS). The phonetic symbols should unambiguously represent the phonetic nature of the read text. The Czech TTS system is described in [1].

Nowadays the most widely used method combines the lexicon-based and the rule-based phonetic transcription. Some words are converted directly using a pronunciation lexicon, whereas phonetic rules are applied on the words that are not contained in the lexicon. The primary disadvantages of this approach are the large memory requirements to store the lexicon, and the need of time-consuming creation of a large lexicon. Thus, it looks like a good idea to use only phonetic rules, however the accuracy of phonetic transcription can be then dropped out. Moreover, some words in each language (exceptions to phonetic transcription rules) cannot be transcribed according to rules. Such words and their pronunciation still have to be stored in the exception lexicon. Examples of exceptions are usually personal and geographic names and foreign words. Our method tries to reduce the large lexicon memory requirements because these lexicons cannot be completely eliminated. Phonetic transcription is generally language dependent because the sounds which compose speech are different in different languages. In this paper we propose a solution of effective large lexicon storing which can be used for any language.

Our aim is to build a system which can be used to perform phonetic transcription. We will use a lexicon for transcription but we want to reduce the memory demands. The whole system should be language independent and designed for real-time operation. The system lexicon will be primarily used to transcribe the exceptional words.

In the following Sections a more detailed description of the phonetic transcription system is given. In the Section 2, the lemmatization technique (lemmatization) and its use for phonetic transcription are presented. The lemmatization algorithm, its implementation, and the important data structures are also described. Section 3 contains the experimental results and Section 4 summarizes the paper.

2 Technique Description

As noted above, our intention was to create a language-independent system for phonetic transcription of exceptional words. The main goal is to decrease the lexicon memory requirements, so that the phonetic transcription algorithm is fast enough to operate in real TTS applications. As it will be shown, the lemmatization technique can represent an advisable solution of this task.

2.1 Lemmatization

The lemmatization procedure reduces a group of words with the same stem to one word (called a basic form, base or lemma). The base is usually the canonic word form (for example verb infinitive). Lemmatization means searching for the base. For example in English, we use a possessive s (‘s); so we can derive the word JOHN’S from the word JOHN. If we lemmatize JOHN’S we obtain JOHN. JOHN’S is the only word that we can derive in English from JOHN. In Czech, we can derive from JAN (JOHN) the words: JANŮV, JANOVA, JANOVO, JANA, JANOVI, JANE, JANEM, etc. We can derive new words even from the word JOHN: JOHNŮV, JOHNOVA, JOHNOVO, JOHNA, JOHNOVI, JOHNE, JOHNEM, etc. There is a difference between English and Czech how we can see from the above example. Like other Slavic languages Czech is a high inflection language. For example, a Czech verb has about 30 and more different forms (contrary to English 4 forms of regular verb and 8 forms of irregular verb [2]).

There are two main processes used for derivation of new words in a language: the inflectional and the derivative process. In the inflectional process the words are derived from the same morphological class (for example the form CLEARED and CLEARS of the verb CLEAR) while in the derivative process the words are derived from other morphological classes (CLEARLY). Creation of a new word can be reached by applying a set of derivation rules in both processes. The rules provide adding or stripping prefixes and suffixes to derive a new word form. For example, the Czech negation is created by adding the prefix NE. From this point of view the lemmatization can be regarded as the inverse operation to the inflectional and derivative processes. This approach is advantageous, because we do not need to create the lemmatization rules from scratch, but we can obtain them through the inversion of derivation rules. The derivation rules are a part of the language grammar and therefore they can be easily deduced. The inversion process of derivation rules is described in the next Section.

The lemmatizer based on language morphology consists of three parts:

1. The set of derivation rules.

2. The lexicon of basic word forms with information which derivation rules can be applied for each basic form.

3. The lemmatization algorithm for finding a basic form stored in the lexicon to the given (generally non-basic form) word.

The set of derivation rules is a set of if–then rules (for example a simple possessive derivation rule is: if the last character of the word is not S, then add S). The set of rules should cover all morphology events of the given language. The completeness of the lexicon strongly influences the successfulness of lemmatization because the proper basic form can only be found if it is included in the lexicon.

2.2 Lemmatization Algorithm

If we are looking for a base of a given word W then we suppose that one or more derivation rules applied on the base can yield the word W. The equation

$$W = P^{'} + S + U^{'} \qquad (1)$$

describes the derived word W. Letters $P^{'}$, S and $U^{'}$ denote the prefix, the stem and the suffix, respectively. The aim of the algorithm is to find the original basic form

$$B = P + S + U \qquad (2)$$

where P is a prefix and U is a suffix. We cannot simply strip the prefix $P^{'}$ and suffix U', but we have to apply inverse derivation rules (called lemmatization rules) on the word W. The derivation rule can be generally written as:

$$\text{If C then R} \qquad (3)$$

where C is a rule condition and R is a result which is applied on the word if the condition C is true. The general form of the result R is:

$$- S, A \qquad (4)$$

where S is a stripped string and A is an added string. If the condition C is applied on the beginning or the end of the word then we speak about the prefix or suffix rule, respectively. The whole rule has then a form:

$$\text{If C then} - S, A \qquad (5)$$

Now we can induce the inverse form of the derivation rule simply as follows:

$$\text{If A then} - A, S \qquad (6)$$

The condition C plays no role in the inverse rule form (is applied only when all words from the basic form have to be generated). The rule which has no part A in its part R is the stripped-only rule. The stripped-only inversion rules are rules with always true condition C. We have to apply all these inversion rules on each investigated word which significantly decelerates the lemmatization process. Therefore, the stripped-only rules have to be removed from the derivation rule set and their corresponding word forms added to the lexicon directly.

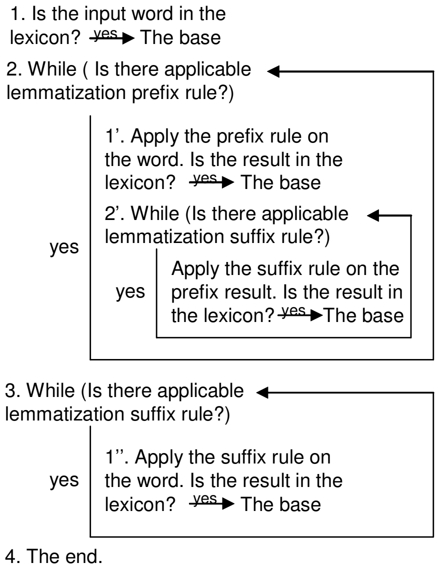
1. Is the input word in the lexicon? yes The base
2. While (Is there applicable lemmatization prefix rule?)
yes
1'. Apply the prefix rule on the word. Is the result in the lexicon? yes The base
2'. While (Is there applicable lemmatization suffix rule?)
yes
Apply the suffix rule on the prefix result. Is the result in the lexicon? yes The base
3. While (Is there applicable lemmatization suffix rule?)
yes
1''. Apply the suffix rule on the word. Is the result in the lexicon? yes The base
4. The end.

Fig. 1. The lemmatization algorithm for finding the basic form

The algorithm has to go through all inversion (lemmatization) rules and stops when no usable rule remains. This is due to the presence of homonymy phenomenon. The homonymy means that we can derive the identical word form from two or more bases (for example the Czech word TANCÍCH can be derived from the bases TANK or TANEC). The correctly working lemmatizer has to find all bases which the word form can be derived from.

The algorithm speed depends on the speed of searching the base in the lexicon and searching in the lemmatization rule set. Therefore a hash table has been employed as a data structure for storing both the lexicon and the rules. The hash table is theoretically the fastest way of data access among other possible solution as AVL trees, B trees, tries, etc.

2.3 Lemmatizer Implementation

Our implementation is based on language morphology (inversion derivation rules) and uses the above mentioned lemmatization algorithm for finding the word basic form. The lemmatizer uses two files: the file of the derivation rules and the file containing the lexicon of basic forms of all exceptional words. If these two language dependent files are assumed as an input of the system then the lemmatizer is language independent because we can use appropriate input files for any language.

The structure of the both files is compatible with the structure of files used by the Linux spellchecker program Ispell. Its lexicon file and the rule file are available on the internet and hence our system can be easily tested by them. The derivation rules are clustered into the groups in the rule file. Each group has been assigned by its identification flag which is also stored in the rule file.

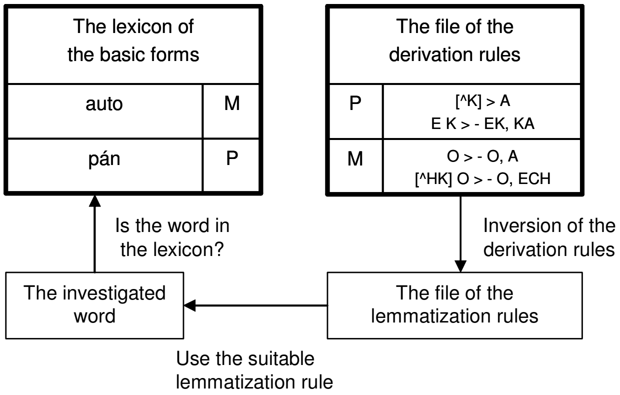
The lexicon of the basic forms
auto
M
pán
P
The file of the derivation rules
P
[^K] > A
E K > - EK, KA
M
O > - O, A
[^HK] O > - O, ECH
Is the word in the lexicon?
Inversion of the derivation rules
The investigated word
The file of the lemmatization rules
Use the suitable lemmatization rule

Fig. 2. The scheme of the lemmatizer and the input files

The bases are stored in the lexicon file together with their flags that determinate which derivation rules from the rule file are applicable for a given base. Any numbers of flags can be assigned to each base in the lexicon file. We use the information, which rules are applicable on a given base to find the right basic form. After the basic form has been found in the lexicon it is checked if the lemmatization rules used to find the base are permitted, i.e. if their flag is among the flags of the base.

2.4 Phonetic Transcription Using the Lemmatizer

The lemmatizer input files have to be modified to contain the information on phonetic transcription.

The lexicon of the basic forms		
The base	The phonetic transcription	The flag
auto	auto	M
pán	paan	P

The lexicon of the basic forms	
P	the rule / the phonetic transcription [^K] > A / [^K] > A
M	O > - O, Á / O > - O, AA [^HK] O > - O, ECH / [^HK] O > - O, ECH

Fig. 3. The modification of the input files

A new extra column with phonetic transcription of the base has been added and stored in the lexicon file. Furthermore, phonetic transcription of each derivation rule (separated by slash) has been added in the rule file. The lemmatization rules have been created by inversion of the derivation rules (only the part before the slash is inverted). The lemmatizer now searches the input word base in the first lexicon column. If the base is found, then its phonetic transcription from the second lexicon column together with the phonetic transcription (the non-inverted part after the slash = phonetic derivation rule) of each successfully used lemmatization rule are taken, and finally the phonetic derivation rules (i.e. inverse process to the lemmatization just performed now) are applied (in a reverse order to the lemmatization rule sequence) to a phonetic transcription of the base yielding the resultant phonetic transcription of the input word.

3 Experimental Results

The lemmatizer has been tested on lexicons and rules for several different languages. For testing we could use Ispell files because the structure of the lemmatizer input files is compatible with Ispell file structure.

Table 1. The experimental results

The language	English	Czech	Finish
The size of the file with lexicon of the bases	894,740 B	2,628,863 B	926,679 B
The number of the words	83,711	172,866	88,392
The size of the file with all the word forms	1,527,402 B	46,712,474 B	99,258,202 B
The number of the words	135,898	3,201,163	5,048,861
The size of the file with lexicon hash table	1,248,584 B	2,988,653 B	1,347,929 B
The number of derivation rules	46	2,546	18,618

If we compare the full lexicon size F (the file with all word forms) with the size B of the lexicon hash file containing only the base form of words, then the compression ratio B/F * 100 is 81.7 % for English, 6.4 % for Czech, and 1.4 % for Finnish. The proposed method is thus advisable especially for high inflectional (Czech) and agglutinative (Finnish) languages.

The lemmatizer functionality has been tested only on Czech. A lemmatizer input is a hash table of lemmatization rules, a lexicon hash table and a test file, in which all words that can be derived from all bases of the lexicon are stored. The output file comprises the input file word in the first column, and all bases found by the lemmatizer for the put word in the next columns. This file was compared with a reference file (the reference file contains all word forms to each word base). The reference file includes an (generally non-basic) word form in the first column and its base in the second column. If this base occurred among the bases stored in the lemmatizer output file, then the word was lemmatized correctly. 280 words were lemmatized incorrectly. The error analysis showed that these errors were caused by the errors in the rules (mismatch between the condition and the stripped string). After these rules had been repaired then all words were lemmatized correctly. This result means that the lemmatizer using the inverse derivation rules works correctly for all tested (basic and derived) words. The speed of the lemmatization was about 19,230 words / s on Pentium 4 2.5 GHz, 512 MB RAM.

4 Conclusion

In this paper, the system which can be used to phonetic transcription of the exceptional words has been presented. Our main goal has been to lower memory requirements of the exceptional lexicon. The solution based on lemmatization has been chosen and tested. The lemmatizer has been modified to deal with phonetic transcription. The lemmatizer is based on language morphology and uses the lexicon of word basic forms and inversion of the derivation rules to acquire the lemmatization rules which are essential for finding the word bases. The lemmatizer functionality has been tested on Czech files and all the derived words have been successful lemmatized. We measured the memory saving for three different languages: analytical – English, high inflectional – Czech and agglutinative – Finnish. The best result has been achieved for Finnish (98.4% of memory saving) and for Czech (93.6%) while the relatively low result has been obtain for English (18.3 %). It is because that only about 46 derivation rules are sufficient for English contrary to 2,546 rules for Czech and 18,618 rules for Finnish.

References

1. Matousek J. and Psutka J.: ARTIC: A New Czech Text-to-Speech System Using Statistical Approach to Speech Segment Database Construction.-In: The Proceedings of the 6th International Conference on Spoken Language Processing ICSLP2000, vol. IV. Beijing, China, 2000, pp. 612-615.
2. Sedlacek, R., Smrz, P.: Automatic Processing of Czech Inflectional and Derivative Morphology. FIMU Report Series, Faculty of Informatics, Masaryk University, Czech Republic (2001)
3. Hajic, J.: Statistical Nature Language Modeling and Automatic nature language analyze http://ufal.mff.cuni.cz/publications/year2001/slovko1.doc (only in Czech)
4. Strossa, P.: Czech Lemmatizer. Why and How?. *Computerworld*, vol. 13, 2002, no. 29, pp. 9–11 (only in Czech)
5. The online manual for the program *ISPELL* http://h30097.www3.hp.com/demos/ossc/man-html/man4/ispell.4.html#lbAB