Grapheme-to-Phoneme conversion, a knowledge-based approach

Torstensson, N.

journal: Proceedings of Fonetik, TMH-QPSR
volume: 44
number: 1
year: 2002
pages: 117-120

http://www.speech.kth.se/qpsr
Grapheme-to-phoneme conversion, a knowledge-based approach.

Niklas Torstensson

Dept. of Languages, Högskolan i Skövde

Abstract

This paper reflects the results of an ongoing project at Högskolan i Skövde, aimed at the creation of a system for grapheme-to-phoneme conversion for Swedish, from a knowledge-based approach. The focus lies on development and implementation of an algorithm for parsing orthographic text, and phonetic rules for the transcription.

Introduction

This paper deals with the creation of a knowledge-based grapheme-to-phoneme conversion system for Swedish. The work can be seen as having four main points of focus including (1) development of a parsing algorithm to handle textfiles containing orthographic text in Swedish, (2) a method to represent phonological rules in a form adapted to computation, (3) development of phonological rules for transcription of orthographic text into a suitable phonetic representation, and (4) testing and evaluation of the system. The algorithm and the rules have been implemented, tested and evaluated using a software application written for this purpose.

The trend in TTS-research has been the use of statistical and data-oriented methods instead of this project’s more knowledge-oriented approach. The grapheme-to phoneme conversion has been solved by using huge databases and neural networks to map the spelling of words to a string of phonetic symbols representing the pronunciation (e.g. Daelemans & Bosch -96).

The motive for undertaking this project is as multi-faceted as the research field itself. There are however three main motives. The first is to create an algorithm for handling and parsing the input-text to make it possible to apply transcription rules to it. This step includes some simple text normalization.

The second motive is the most challenging and consists of writing the actual rules to govern the transcription from orthographic text in Swedish to a phonetic representation. This step includes finding a form to represent the rules in a computational-friendly way. This because the prevalent standard for phonological transcription is working well for writing and reading on paper, but is ill-adapted to computation.

Motive number three is to find a suitable representation for the output, in such a way that it must be adapted for use in further steps in the creation of a complete text-to-speech system.

Goals

The goal is to create a grapheme-to-phoneme algorithm that can handle orthographic text written in Swedish sufficiently well. A realistic expectation of performance is to achieve a correctness of 80% on word-level and better than 90% correct on phoneme level. This is a realistic expectation as it has been achieved in knowledge-based systems for other languages. (Bernstein & Pisoni -80)

It must be kept in mind that the system is limited in function, and thus cannot be expected to handle anything outside standard Swedish phonotax. Things that fall outside this scope, as some proper names, acronyms, some loanwords and technical terms are not expected to be correctly transcribed.

Method

The first step is to decide the different forms of representation, i.e. on what form the input, the rules and the output should be in order to
facilitate future work. When this is decided, the parsing algorithm must be designed and implemented in a way that it can handle the specific features of any random text in Swedish together with the transcription rules. The format of the rules must be on a form that can express all features of standard phonetic rules but, as opposed to these, be adapted to computation. An application to test the function of the system is needed, as the method by necessity is iterative and changes between the construction of rules and the testing and evaluation of them. This application is programmed object-oriented and event-driven. It is implemented in Delphi, and it communicates with a relational database written in MS Access where the transcription rules are stored. The method has thus been iterative, changing between the elaboration of rules, implementation of these rules, testing them using the application program and correcting errors.

**System encoding**

**General idea**

The general thought behind the architecture takes a stance in phonotax and graphemes rather than in phonetics or computation. The basic question is what knowledge is required to determine how a certain letter is pronounced according to Swedish phonotax. The obvious answer must be found in the given context. There are some very general phonotactic rules that apply to Swedish. For instance, a long vowel is followed by a short consonant, and a short vowel is followed by a long consonant. This is well known basic knowledge, but only states facts about duration, not quality of a sound. To state facts about the sound quality connected to a grapheme, it is necessary to see what precedes and succeeds that grapheme.

To cope with this, a database is used to store contextual information about graphemes in different contexts. This information is formulated as rules for the different phonological alternatives that can apply to a grapheme depending on what context it is found in. This is implemented by specifying what is in the position before the grapheme at hand (precontext) and what follows after it (postcontext). This method requires several rules for most graphemes, with the exception of some consonants that get the same transcription regardless of their context.

The specific mapping between the graphemes and the phonetic transcription is based on the individual graphemes’ place in a context. Each grapheme is classed as belonging to either the class consonants (K) or vowels (V). The graphemes classed as vowels are then given an index number for articulatory position, in accordance with the traditional IPA vowel chart. These index numbers are set as coordinates in a grid placed over the IPA vowel chart (International Phonetic Association -99).

**Architecture**

The system can be viewed as consisting of separate modules and sub-modules. The three main modules are the algorithm module, the evaluation application and the rule database. The central parts are the algorithm and the rules, for practical reasons stored in a relational database.

![Figure 1.](image-url)

**The evaluation application**

The evaluation application is, as mentioned earlier, programmed to be event-driven and object-oriented. It consists mainly of functions to help and assist in development and evaluation of the transcription rules. The functionality of this module is, as shown in fig. 6, to handle the incoming orthographic textfile by opening the text-file, normalise it and split the text into an array of words. Other functions are to communicate with the RDB keeping the set of rules updated, and to handle the output in a way that facilitates interpretation of the result. The simple GUI is implemented for several reasons. As it is a necessary part of the development to keep track of what transcription rule is used for each phoneme, this application makes it possible to see all rules used for each transcribed lexeme.
The rules used to transcribe the lexeme /skydda/ are seen in the Rules-field of the application GUI, and they are presented with hyphens as delimiter. In other words – the input lexeme /skydda/ is SAMPA-transcribed as [SYda] using the rules:

- skV(*,>=2)%-yKX(X=$)%-dd%-X(X=$)Ka%

The Rule Database (RDB)

The RDB is a relational database implemented in Microsoft Access, and it contains information about all Swedish graphemes, the phonotactic rules, categorization into vowels or consonants and some articulatory facts about the vowels. To cope with some special features of Swedish spelling / articulation concerning some conjunctions, a part of the RDB works as a very small lexicon. The lexicon part of the DB also deals with numerals from 0 to 10. For the system to be able to cope with larger numerals, a separate parsing algorithm along the same principles will be needed just to transcribe these. This could be an idea for a future research project along the same lines as this.

The parsing algorithm

The requirement specification when building the parsing algorithm was that it must deal with more than one grapheme at a time in order to deal with contextual restraints. Since a Swedish phoneme can consist of one, two, or a maximum of three graphemes realized as one phoneme, the algorithm has to take three graphemes at a time into consideration. The algorithm works as shown in Fig. 2.

Step one is splitting the input text into an array of words, and at the same time performing text-normalization. This means to clean the text from non-printing characters as multiple white-spaces and carriage-returns. Step two is to turn the first word into an array of chars, setting a startvalue to 1 and endvalue to 3. These first three chars are then compared to the RDB. If a matching rule is found, the SAMPA-transcription goes to a temporary word-string, and if not end-value is decreased with 1, leaving only one char to compare to the RDB. This corresponding transcription is sent to the temp word-string, start value is set to 1 and end-value is set to 3, and the same procedure is repeated.

This iterates for each word until end of word, and the temp word-string is sent to a temp output-string. When end of file is reached, the resulting temp output-string is sent to the temporary output-string and stored. The source code for the algorithm can be seen in Appendix 2.

Figure 2 – Algorithm flowchart

Results

The system was tested on a number of different texts of varying kinds. The initial testing was performed using short excepts from Swedish newspapers on the web. To get more reliable and statistically valid results, longer texts were used. Test text no. 1 is the lyrics from an LP-album with text of a more poetic style. The reason for using a text of this type is to find the same combination of phonemes in several
different contexts, since the text is rhymed. It also contains a significantly higher occurrence-rate of compound words consisting of more than two constituents. The text file used consists of 2854 words including punctuation marks. Without punctuation marks the sum of words is 2807, and the total sum of unique words is 999. Out of these 999 unique words, 791 were correctly transcribed and 208 had some error in the transcription. This gives a correctness level of 79.2%.

The second input text file, text no. 2, is an excerpt from a Swedish fiction book consisting of 11476 words including punctuation marks. Without punctuation marks the sum of words is 10222, and the sum of unique words is 2684. Out of these, 606 had some error in the transcription, which gives a correctness level of 77.4%.

A view of the results calculated on phoneme level gives that out of a total of 11482 graphemes in text no. 1, 10936 are correctly transcribed and 546 graphemes are transcribed into the wrong phoneme. This gives a performance of 95.2%.

As for the second text, out of 49744 graphemes in text no. 2, only 721 graphemes were transcribed into a wrong phoneme. This gives a correctness of 98.6% on phoneme level, which is a bit over what was initially expected.

Table 1. Results

<table>
<thead>
<tr>
<th>Test text No. 1</th>
<th>Test text No. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of words</td>
<td>2854</td>
</tr>
<tr>
<td>No of unique words</td>
<td>999</td>
</tr>
<tr>
<td>No. correctly transcribed</td>
<td>791</td>
</tr>
<tr>
<td>No. of erroneous</td>
<td>208</td>
</tr>
<tr>
<td>Correct rate – word level</td>
<td>79.2 %</td>
</tr>
<tr>
<td>No of graphemes</td>
<td>11482</td>
</tr>
<tr>
<td>No of correctly transcribed</td>
<td>10936</td>
</tr>
<tr>
<td>No of erroneous</td>
<td>546</td>
</tr>
<tr>
<td>Correct rate – phoneme level</td>
<td>95.2 %</td>
</tr>
</tbody>
</table>

These results are close to and, on phoneme level, over the initial expectations of performance level and in accordance with earlier results along the same lines.

Error types

The transcription errors can be divided into different sub-types to explain the results more accurately. The main error-types are

I. Quantitative errors - where a vowel is transcribed as long instead of short, or vice versa.

II. Qualitative errors – where a vowel is transcribed as rounded instead of un-rounded or vice versa.

III. Character error – where a wrong character is used in the transcription.

The most frequent of these error-types are of type I, the quantitative errors, where the length of a vowel is transcribed wrongly. 48.8% of the total errors are of the type that a long vowel is transcribed as short, and in 21.8% of the cases a short vowel is transcribed as long. Type II errors, of a qualitative kind occurs in 4.4% of the errors. That a grapheme is transcribed as the wrong phoneme, error type III, occurs in 4.0% of the erroneous transcriptions. The remaining 20.6% consists of words with multiple errors, i.e. two or more of the above errors in the same word.

The type II errors are mostly associated with transcription of the grapheme /u/ and distributed over different contexts. The reason for this appears to be that no general rule can be written to cover all possible aspects of vowel quality for this grapheme in Swedish. Type III errors appear in two different contexts. The first is in foreign loanwords which do not follow Swedish phonotax, i.e. words like “trenchcoat” and “sherry”. The second context is in joints between lexical entities of compound words where grapheme combinations can appear that gets wrongly transcribed.

References

