A Language-Independent, Data-Oriented Architecture for Grapheme-to-Phoneme Conversion

Walter Daelemans and Antal van den Bosch*

`Proceedings ESCA-IEEE speech synthesis conference, New York, September 1994`

Abstract

We report on an implemented grapheme-to-phoneme conversion architecture. Given a set of examples (spelling words with their associated phonetic representation) in a language, a grapheme-to-phoneme conversion system is automatically produced for that language which takes as its input the spelling of words, and produces as its output the phonetic transcription according to the rules implicit in the training data. This paper describes the architecture and focuses on our solution to the *alignment problem*: given the spelling and the phonetic trancription of a word (often differing in length), these two representations have to be aligned in such a way that grapheme symbols or strings of grapheme symbols are consistently associated with the same phonetic symbol. If this alignment has to be done by hand, it is extremely labour-intensive.

1 Introduction

Grapheme-to-phoneme conversion is an essential module in any text-to-speech system. Various language-specific sources of linguistic knowledge (at least morphological and phonotactic) are taken to be necessary for implementing this mapping with reasonable accuracy. Accordingly, an expensive linguistic engineering phase is involved in developing text-to-speech systems. In this paper we describe an implemented grapheme-to-phoneme conversion architecture that allows data-oriented induction of a grapheme-to-phoneme mapping on the basis of examples, thereby eliminating this knowledge acquisition bottleneck.

Input to our system is a set of spelling words with their associated pronunciation in a phonemic or phonetic alphabet (the training data). Spelling and pronunciation do not have to be aligned. The phonetic transcription can be taken from machine-readable or scanned dictionaries, or from automatic phoneme recognition. The words may represent text in context (when effects transgressing word boundaries

*ITK, Tilburg University; PO Box 90153, 5000 LE Tilburg The Netherlands; Phone: +31 13 663070; Fax: +31 13 662537; E-mail: walter.daelemans@kub.nl

have to be modeled) or isolated words. Output of the system is a grapheme-to-phoneme conversion system which takes as its input the spelling of words, and produces as its output the phonetic or phonemic transcription according to the rules implicit in the training data.

The architecture has a number of desirable properties:

1. It is *data-oriented.* The output system is constructed automatically from the training data, thereby effectively removing knowledge acquisition bottlenecks. Linguistic solutions to the problem need considerable handcrafting of phonological and morphological datastructures, analysis and synthesis programs.

2. It is *language-independent and reusable.* Versions of the system for French, Dutch and English have been automatically constructed using the same architecture on different sets of training data. In linguistic approaches, the handcrafting has to be redone for each new (sub)language.

3. It achieves a high accuracy. Output of the Dutch version has been extensively compared to the results of a state-of-the-art 'hand-crafted', linguistic system. The data-oriented solution proved to be significantly more accurate in predicting phonetic transcriptions of previously unseen words. Output of an American English system generated by the architecture and based on the Nettalk data was more accurate than Nettalk, Memory-Based Reasoning, and other inductive solutions to the problem (Daelemans & van den Bosch, 1993; Van den Bosch & Daelemans, 1993).

2 Design of the System

The system consists of the following modules: (i) *Automatic alignment*: spelling strings and phonetic strings have to be made of equal length in order to be processed by the other modules. (ii) *Automatic training set compression*: part of the training data is represented in a compact way using trie structures. (iii) *Automatic classifier construction*: using the compacted training data and similarity-based reasoning techniques enriched with techniques from information theory, a classifier is constructed that extrapolates from its memory structures to new, unseen input spelling strings. Module (i) will be discussed extensively in the next section.

(ii) Automatic training set compression can be seen as optimized, generalized lexical lookup. The training set is compressed into a grapheme-to-phoneme conversion *trie.* The main strategy behind this compression is to dynamically determine which left and right contexts must minimally be known to be able to map a single grapheme to its corresponding phoneme with absolute certainty (in the training corpus). Generalisation is achieved because of the fact that unknown words usually contain known substrings of graphemes. Finding a phonemic mapping of a grapheme is done by a search through the trie taking into account a variable amount of context. The order in which the context graphemes are added to the trie search is not randomly determined, but is computed using the concept of *Information Gain* (IG). This ordering method is used in a similar way in C4.5-learning (Quinlan, 1993). The main difference with C4.5-learning is the fact that our model computes the expansion ordering only once for the complete trie, whereas in C4.5-learning the ordering is computed at every node.

(iii) Automatic classifier construction is achieved by combining the trie compression with a form of similarity-based reasoning (based on the k-nearest neighbour decision rule, see e.g. Devijver & Kittler, 1982). During training, a memory base is incrementally built consisting of *exemplars*, which in the case of grapheme-to-phoneme mappings consist principally of a strings of graphemes (one focus grapheme surrounded by context graphemes) with the associated phonemes and their distribution (as there may be more phonemic mappings to one graphemic string). During testing, a test pattern (a graphemic string) is matched against all exemplars. If the test pattern is in memory, the category with the highest frequency associated with it is used as output. If it is not in memory, all memory items are sorted according to the similarity of their pattern to the test pattern. The (most frequent) phonemic mapping of the highest ranking exemplar is then predicted as the category of the test pattern. Daelemans & Van den Bosch (1992) extended the basic IBL algorithm by introducing Information Gain as a means to assigning different weights to different grapheme positions when computing the similarity between training and test patterns (instead of using a distance metric based on overlap of patterns).

The Trie Search algorithm is combined with the Information Gain-aided k-nn technique in the following way: Trie Search succeeds only when a completely matching path can be found up to the node where the phonemic mapping becomes unambiguous. New, unseen test words may very well contain graphemic substrings that are not present in the training data. In those cases, Trie Search will fail somewhere halfway. In our architecture, information-gain extended k-nn is used on a memory base of exemplars when Trie Search fails.

Components (ii) and (iii) of the system, as well as its evaluation in comparison to linguistic, knowledge-based solutions and to connectionist and alternative data-oriented solutions have been reported in detail previously in Van den Bosch and Daelemans (1993) and Daelemans and Van den Bosch (1993). In this paper we will focus on our as yet undocumented solution to the alignment problem implemented in module (i).

3 Automatic Alignment

The alignment algorithm operates on any data set of words associated with their transcriptions. The algorithm attempts to equal the length of a word's spelling string with its transcription. This is done by adding *null* phonemes to the transcriptions. Instead of just concatenating the required number of nulls at the end of the transcription, nulls have to be inserted in the transcription at those points in the word where a letter cluster maps to one phoneme. The word 'shoe'-/Su/, for example, contains two letter clusters, 'sh' and 'oe', both mapping to one phoneme. A possible alignment that would be at least intuitively correct would then be the transcription /S - u -/. The transcription /S u - -/ on the other hand would definitely not be intuitively correct. The first part of the algorithm automatically captures these typical letter-phoneme associations in an association matrix. Each spelling string is aligned to the left with its (possibly shorter) transcription. For each letter, the score of the phoneme that occurs at the same position in the transcription is incremented; furthermore, if a spelling string is longer than its transcription, phonemes which precede the letter position are counted as possibly associated with the target letter as well. In the example of 'shoe', for each letter, three phonemes receive a score increase (underscores indicate word boundaries and do not count as phonemes):

letter	focus-2	focus-1	focus
s	_	_	S
h	_	S	u
o	S	u	_
e	u	_	_

Although a lot of noise is added to the association matrix by including associations that are less probable, the use of this association window ensures that the most probable associated phoneme is always captured in this window. The score of the phonemes is not increased equally for all positions: in the present implementation, the focus phoneme receives a score increase of 8; the phonemes to the left receive a score increase of 4, 2, and 1 respectively; phonemes situated further in the string do not receive any score. Other values for these weights result in slightly (but not significantly) worse results. When all words are processed this way, the scores in the association matrix are converted into probabilities.

The second part of the alignment algorithm generates for each pair of unaligned spelling and phoneme strings all possible (combinations of) insertions of null phonemes in the transcription. For each hypothesized string, a total association probability is computed by multiplying the scores of all individual letter-phoneme association scores between the letter string and the hypothesized phonemic string. The hypothesis with the highest total association probability is then taken as output of the algorithm.

The resulting alignment is not always identical to the intuitive alignment applied by human coders. To test its efficacy, we compared classification accuracy of the complete system when using a hand-aligned training set as opposed to the automatically aligned training set. The results indicate that there is no significant difference in classification accuracy: the alignments result in systems that are equally accurate. The resulting trie is on average about 3% larger with the automatically generated alignment, however.

4 References

Bosch, A. van den and W. Daelemans, 'Data-oriented methods for grapheme-to-phoneme conversion.' *Proceedings of the Sixth conference of the European chapter of the ACL*, ACL, 45-53, 1993.

Daelemans, W. & A. van den Bosch (1992). Generalization performance of backpropagation learning on a syllabification task. In M. Drossaers & A. Nijholt (Eds.), *Proceedings of the 3rd Twente Workshop on Language Technology*. Enschede: Universiteit Twente, 27-37.

Daelemans, W. and A. van den Bosch. 'TABTALK: Reusability in Data-oriented grapheme-to-phoneme conversion.' *Proceedings of Eurospeech*, Berlin, 1459-1466, 1993.

Devijver, P.A. & J. Kittler (1982). *Pattern recognition. A statistical approach.* London: Prentice-Hall.

Quinlan, J.R. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.