Proceedings of IEEE ICASSP'93, Vol. II, pp. 199-202, Minneapolis, Minnesota, USA, April 27-30, 1993

AN EFFICIENT WAY TO LEARN ENGLISH GRAPHEME-TO-PHONEME RULES AUTOMATICALLY

Kari Torkkola

IDIAP (Institut Dalle Molle D'Intelligence Artificielle Perceptive)
C.P. 609, CH-1920 Martigny, SWITZERLAND
email: karit@idiap.ch

ABSTRACT

We present an efficient way to learn automatically grapheme-to-phoneme mapping rules for English by using Kohonen's concept of Dynamically Expanding Context. This method constructs rules that are most general in the sense of an explicitly defined specificity hierarchy. As the hierarchy, we have used the amount of expanding context around the symbol to be transformed, weighted towards the right. To apply this concept to English text-to-speech mapping, we have used the 20008-word corpus provided in the public domain by Sejnowski and Rosenberg, that was also used in the NETTALK-experiments. Phoneme-level mapping accuracies of 91 per cent with data not used in training demonstrate that the Dynamically Expanding Context is able to capture quite efficiently the context-dependent relationships in the corpus.

1 INTRODUCTION

The problem addressed in this paper is automatic learning of grapheme-to-phoneme mapping rules. We present an efficient way to learn these for English by using Kohonen's concept of Dynamically Expanding Context (called the DEC hereafter) [9]. The purpose of the paper, however, is not to construct any actual text-to-speech system but to study and demonstrate the learning and generalization capabilities of this simple, yet powerful machine learning scheme.

The DEC is an algorithm to construct a set of context-dependent transformation rules from examples of source and target strings. So far, this concept has been in a reverse use; to transform erroneous phonemic transcriptions, produced by an automatic speech recognition system, into orthographically correct text [9] [16]. In that task the DEC has been used to learn the structure of systematic, mostly coarticulation-based errors. The DEC has also been applied to predict future values for a time series, especially to automated composing of music [10].

The task that we now apply the DEC to, that is, transformation of English text into phonemes on the basis of local letter context alone, is inherently impossible. In addition to text content, pronunciation is determined by several other points based on linguistics, such as the part-of-speech of the word, morphological structure of the word, and etymology. Simple illustrative examples of the first case are *read*, where the pronunciation depends on the tense of the verb, *intimate* (adjective or verb), or *contrast* (noun or verb). An example, where morphology affects the pronunciation over a long duration, is the word pair *photograph / photography*, where the suffix "*y*" affects the pronunciation of the first vowel (among other things).

Due to its difficulty, there has been a lot of interest in this specific task as a test bench for machine learning algorithms, both connectionist and not. Examples of this kind of work include [4], [3], [12], [13], and [14]. Most of this work tends to ignore any additional knowledge about the problem domain but the examples. This is, of course, a good viewpoint when one is demonstrating the validity of a machine learning method, or when no such knowledge does exist. We, however, introduce a small piece of knowledge about the problem domain, at this point. We do it by making the obvious assumption, that an important thing in determining the pronunciation of a letter is not only the local text context per se, but also its distance to the letter whose pronunciation is at consideration. This paper demonstrates how this assumption, combined with a simple decision tree building algorithm can produce convincing results.

A necessary component in a text-to-speech system is also stress assignment [8]. Since this is even more dependent on other issues but local letter context [2], we have decided to concentrate only on grapheme to phoneme conversion in this work.

In a working text-to-speech system, it might be necessary to construct or at least complete the set of text-to-phonemes rules by hand to make it as complete as possible, but it is a tedious task as noted in [5]. In contrast, there are many domains where machine learning approaches are the only viable ones because of no pre-collected or pre-existing knowledge of the domain.

The rest of this paper is organized as follows. Section 2 gives a brief overview of the DEC-algorithm and discusses some of its merits. In Section 3 we present the experiments with the NETTALK-corpus. Finally, in Section 4 we draw some conclusions and compare this work to that of others.

2 DYNAMICALLY EXPANDING CONTEXT

The central idea behind the DEC is to derive unique symbol to symbol(s) transformation rules from two streams of symbols: the source stream, and the desired target stream [9]. The DEC constructs rules that are most general in the sense of an explicitly defined specificity hierarchy. As the hierarchy, we have used the amount of expanding context around the symbol to be transformed. For other kind of tasks, different specificity hierarchies, such as focusing resolution in representation, are also possible [15].

The algorithm starts at the most general level, by creating initial context-free transformation rules, that in this are case single letter $\Rightarrow$ phoneme -rewrite rules. Now, these initial rules will certainly conflict each other, which means that two rules with the same condition part may have different productions. These cases, originating from different words, are specialized by expanding the discrete symbolic context around the initial condition parts by just a sufficient amount to make the rules unique. The rules will then end up having condition parts with variable lengths. Two slightly different examples of the expanding context around letter '*t*' from word '*abbreviation*' are displayed in Table 1.

context level	symmetrical expansion	right weighted expansion
0	`t`	`t`
1	`ti`	`ti`
2	`ati`	`tio`
3	`atio`	`atio`
4	`iatio`	`ation`
5	`iation`	`ation_`

Table 1. *Examples of expanding specificity hierarchies.*

This least commitment principle guarantees that the generality of the rules will be optimum, and still, all the conflicts within the training cases are solved. The latter would, of course, be the case only if we allowed the context expansion procedure to continue without limits, and if such a far context were anymore meaningful to the rule. In practice, however, the maximum amount of context can be limited, when the remaining conflicts are handled by simple statistical procedures described later in this section.

The set of rules will be stored as an efficient tree structure, which is ordered according to the generality of the rules. The recall phase is thus plain tree ascend; selecting branches to follow according to the gradually expanding context of the letter to be transformed, and outputting the production (the phoneme(s)) found at the leaf. This means that most general rules are accessed first moving to more specific rules only when the general ones are indicated to be conflicting ones. This is in contrast to the scheme presented in [6], in which one begins the search at the most specific end.

The efficiency of the DEC in the learning phase stems from the fact that the training material does not have to be scanned through even as many times as there are context levels allowed. In fact, only a half of this amount is sufficient [9]. In addition, a single final scan is required, the purpose of which is to produce majority estimates for the production part of each conflicting rule.

If a leaf is reached during the tree search, the letter to be transformed is replaced by the production. However, if the search terminates before reaching a leaf, a novel context has been encountered. In this case we have ended up at an intermediate node, which has no branch according to context expansion that ought to be done next. These kind of cases can be handled by the following procedure. For every intermediate node of the tree, that is, for every conflicting rule, estimates of the most probable productions are constructed. To this end, all substrings that match the condition part of this node are searched from the training material, and corresponding productions are enumerated. The most frequent one represents the most probable estimate of the production, given the amount of context. If the search now terminates before a leaf, this estimate at the deepest level reached is used as the production. This procedure applies also to cases where the maximum allowed amount of context was reached, at which point the leaf represents several conflicting training cases.

In addition to pure expansion, other kinds of specificity hierarchies might also be imaginable for the present task. For example, it could be possible to begin with wider grapheme classes, such as vowels and consonants, then specifying to plosives, nasals, front vowels, back vowels, etc. This specifying could be carried out completely before proceeding to context expansion.

Some learning schemes treat all the context with equal importance. An example of this case is the NETTALK, where all information in the context window is treated the same way. The hope is that the learning procedure would find the correct weighting without any a priori information about the relative importance of the context to the mapping. The DEC is able to concentrate on the most important aspect, as defined by the specificity hierarchy, which is partly responsible in making the DEC as efficient as it is.

3 EXPERIMENTS WITH THE CORPUS

To apply the concept of Dynamically Expanding Context to English text-to-speech mapping, we have used the 20008-word corpus *(Miriam Webster's Pocket Dictionary, 1974)* provided in the public domain by Sejnowski and Rosenberg, that was also used in the NETTALK-experiments [13]. The words of the corpus are already aligned with their phonetic transcriptions. The corpus also contains corresponding stress assignments which were ignored in this work.

As the basic unit for transformation rules, we decided to use the single letter for the simplicity of implementation, although one could have combined together letters functioning as a single unit, such as "*ch*" and "*gh*". A further possibility could be to use vowel or consonant clusters as the basic unit.

First, we made experiments in order to find the best performing specificity hierarchy. This means finding the order, in which local letter context is expanded to make too general conflicting rules more specific. These results are displayed in table 2 for one random division of the corpus into a 2000 word test set and a 18008 word training set. In this experiment, the limit of context expansion was set to ten letters; only the order of expansion was varying. The symmetrical, right starting scheme and the right weighted scheme, corresponding to the 4th and 6th rows of table 2 are also illustrated in table 1. On the basis of this experiment, the scheme corresponding to the last row of table 2 was adopted, in which one always expands two times to the right before expanding one letter to the left. In all of the schemes, the expansion was done one letter at the time.

context expansion scheme	accuracy
to left only	78.1
to right only	86.6
symmetrical, starting from left	89.0
symmetrical, starting from right	90.6
1/3 right, 2/3 left	86.8
2/3 right, 1/3 left	90.8

Table 2. *The performance of different kinds of specificity hierarchies.*

To find out the optimal amount of letter context, we performed two experiments with the training and testing sets as follows:

1. Open test. The data is randomly divided into a 18008-word training set and a 2000-word test set. An average of ten such random splits is presented. The test sets contained no same words.
2. Closed test. Using the whole corpus both as the training material and as the testing material.

These results are displayed in Fig. 1 as a function of the maximum allowed context amount. In the open test, which gives an idea of the generalization capabilities of the DEC, phoneme-level mapping accuracies of 91 per cent were obtained. We can see that four letters of context seems to be the limit after which no significant improvement can be achieved with the test data. Since the expansion scheme was that of the last row of Table 2, context level four denotes three letters to the right and one to the left.

Looking at the curve of the closed test, we can see that the performance on the training data is close to perfect, as long as the context is wide enough. This property of the DEC could be used to include an exception dictionary. A working text-to-speech system could use such a dictionary as the first try, and turn to rules when failing. Since the DEC learns the training data almost perfectly, all of the training data can be considered as the exception dictionary. Using the exception dictionary and the rules could thus be covered by the same procedure.

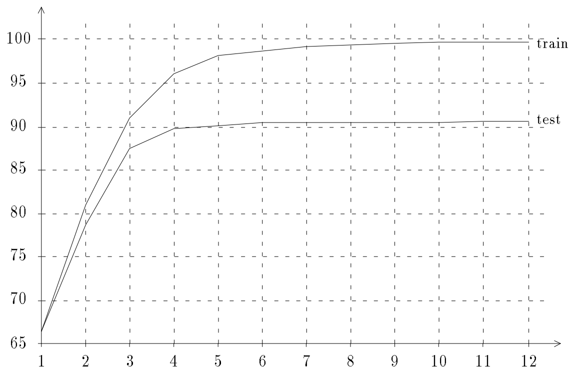

Figure 1. *Transformation accuracy as a function of the maximum amount of context allowed.*

Finally, we wanted to know how the amount of training data affects the results. These results are displayed in Fig. 2 for a test set of 2000 words. The limit of context expansion was ten in this experiment, and the right weighted expansion scheme was used. The behaviour seems to be logarithmic-like. To attain any significant improvements, it seems that one ought at least to double the amount of training data.

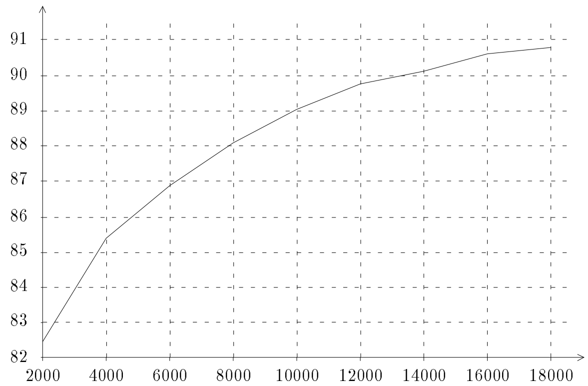

Figure 2. *Transformation accuracy as a function of the amount of training data.*

4 DISCUSSION

We wanted to demonstrate that in learning text-to-phoneme rules from examples, a very simple assumption of the importance of the closeness of local text context (which should be obvious in this case) is clearly a good choice and rivals more sophisticated approaches according to our results. However, we are not trying to claim that this is the only or the best way one must construct such rules; there is a multitude of good approaches in the litterature.

A nice property of the DEC is that the training time is very short, only about 2 minutes on a Sun Sparc-

Station 2. The performance seems to be better than that reported in [13] for the NETTALK, or in [1] for a hierarchical system of high-order neural networks, although the figures are not directly comparable. This demonstrates that the DEC is able to capture quite efficiently the context-dependent relationships in the training data.

A system quite similar in spirit to ours has been reported in [6]. The difference to our scheme is the use of multiple simultaneous context orientations, which leads to the need of devising heuristics how to combine the rules of these different orientations. In our simpler approach there is only one path through the rule hierarchy. Anyway, their results seem to be comparable to ours. Other learning approaches to this task have been reported in [7] and [14].

In [3] error-correcting output codes are combined with ID3 in the same task. Their results are obtained using less training data than in this work. It seems evident, that such codes might be very profitable with the DEC, too. However, it seems that the amount of training data (if available) could compensate for more tedious schemes. On the other hand, one could argue that combining large amounts of training data with the scheme could produce even better results.

In a sense, the DEC is a simplified version of the ID3 or the method of Lucassen and Mercer [11]. In these methods the feature (the letter) to be used next is chosen by information theoretic principles. Our simplification makes the DEC clearly more efficient in the rule construction phase due to the a priori knowledge introduced in the expanding rule specificity hierarchy. On the other hand, this fact makes the DEC less amenable to other tasks as such. One must clearly find a suitable specificity hierarchy for each task separately. This may, however, be a simple and straightforward procedure, as demonstrated in this paper.

References

[1] Z. An, S. Mniszewski, Y. Lee, G. Papcun, and G. Doolen. Hiertalker: A default hierarchy of high order neural networks that learns to read English aloud. In *Proceedings of the IEEE International Conference on Neural Networks*, volume II, pages 221–228, San Diego, July 1988. IEEE.

[2] K. Church. Stress assignment in letter to sound rules for speech synthesis. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP86)*, pages 2423–2426, Tokyo, Japan, April 7-11 1986.

[3] T. G. Dietterich and G. Bakiri. Error-correcting output codes: A general method for improving multiclass inductive learning programs. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, pages 572–577, Anaheim, CA, 1991.

[4] T. G. Dietterich, H. Hild, and G. Bakiri. A comparison of ID3 and backpropagation for English text-to-speech mapping. Technical Report 90-30-4, Oregon State University, Dept. of Computer Science, 1990.

[5] H. S. Elovitz, R. Johnson, A. McHugh, and J. E. Shore. Letter-to-sound rules for automatic translation of English text to phonetics. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(6):446–459, December 1976.

[6] J. Hochberg, S. Mniszewski, T. Calleja, and G. Papcun. A default hierarchy for pronouncing English. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):957–964, September 1991.

[7] D. Klatt and D. Shipman. Letter to phoneme rules: A semiautomatic discovery procedure. *Journal of Acoustical Society of America*, 72:Suppl. 1, S48, 1982.

[8] D. H. Klatt. Review of text-to-speech conversion for English. *Journal of Acoustical Society of America*, 82(3):137–181, September 1987.

[9] T. Kohonen. Dynamically Expanding Context, with application to the correction of symbol strings in the recognition of continuous speech. In *Proceedings 8th International Conference on Pattern Recognition (8th ICPR)*, pages 1148–1151, Paris, France, Oct. 27-31 1986.

[10] T. Kohonen, P. Laine, K. Tiits, and K. Torkkola. A nonheuristic automatic composing method. In P. Todd and D. G. Loy, editors, *Music and Connectionism*, pages 229–242. MIT Press, 1991.

[11] J. Lucassen and R. Mercer. An information theoretic approach to the automatic determination of phonemic baseforms. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP84)*, pages 42.5.1–42.5.4, San Diego, California, USA, March 19-21 1984.

[12] R. Mooney, J. Shavlik, G. Towell, and A. Gove. An experimental comparison of symbolic and connectionist learning algorithms. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI89)*, pages 775–780, Detroit, Michigan, USA, August 20-25 1989.

[13] T. Sejnowski and C. R. Rosenberg. Parallel networks that learn to pronounce english text. *Complex Systems*, 1(1):145–168, 1987.

[14] C. Stanfill and D. Waltz. Towards memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, December 1986.

[15] K. Torkkola. A combination of neural network and low level AI-techniques to transcribe speech into phonemes. In T.Kohonen and F. Fogelman-Soulie, editors, *COGNITIVA-90*, pages 405–416. Elsevier, 1991.

[16] K. Torkkola, J. Kangas, P. Utela, S. Kaski, M. Kokkonen, M. Kurimo, and T. Kohonen. Status report of the Finnish phonetic typewriter project. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN-91)*, pages 771–776, Espoo, Finland, June 24-28 1991.