Proceedings of EACL, Utrecht, 45-53, 1993

Data-Oriented Methods for Grapheme-to-Phoneme Conversion*

Antal van den Bosch and Walter Daelemans
ITK (Institute for Language Technology and AI)
Tilburg University
P.O. Box 90153, NL-5000 LE Tilburg
Tel: +31 13 663070
Email: antalb@kub.nl, walter@kub.nl

ITK Research Report No. 42, March 1993

Abstract

It is traditionally assumed that various sources of linguistic knowledge and their interaction should be formalised in order to be able to convert words into their phonemic representations with reasonable accuracy. We show that using supervised learning techniques, based on a corpus of transcribed words, the same and even better performance can be achieved, without explicit modeling of linguistic knowledge.

In this paper we present two instances of this approach. A first model implements a variant of *instance-based learning*, in which a weighed similarity metric and a database of prototypical exemplars are used to predict new mappings. In the second model, grapheme-to-phoneme mappings are looked up in a compressed text-to-speech lexicon (*table lookup*) enriched with default mappings. We compare performance and accuracy of these approaches to a connectionist (backpropagation) approach and to the linguistic knowledge-based approach.

1 Introduction

Grapheme-to-phoneme conversion is a central task in any text-to-speech (reading aloud) system. Given an alphabet of spelling symbols (graphemes) and an alphabet of phonetic symbols, a mapping should be achieved transliterating strings of graphemes into strings of phonetic symbols. It is well known that this mapping is difficult because in general, not all graphemes are realised in the phonetic transcription, and the same grapheme may correspond to different phonetic symbols, depending on context.

It is traditionally assumed that various sources of linguistic knowledge and their interaction should be formalised in order to be able to convert words into their phonemic representations with reasonable accuracy. Although different researchers propose different knowledge structures, consensus seems to be that at least morphological and phonotactic knowledge should be incorporated in order to be able to find morphological and syllable structure. These structures are deemed necessary to define the proper domains for phonological and phonetic rules. As a typical architecture for grapheme-to-phoneme conversion in Dutch, consider the modules

*This report is a preprint of a paper to be published in the Proceedings of the 6th Conference of the European Chapter of the ACL, Utrecht, April 1993.

in [Daelemans, 1988] shown in Figure 1. It contains most of the traditional datastructures and processing components proposed by computational linguists.

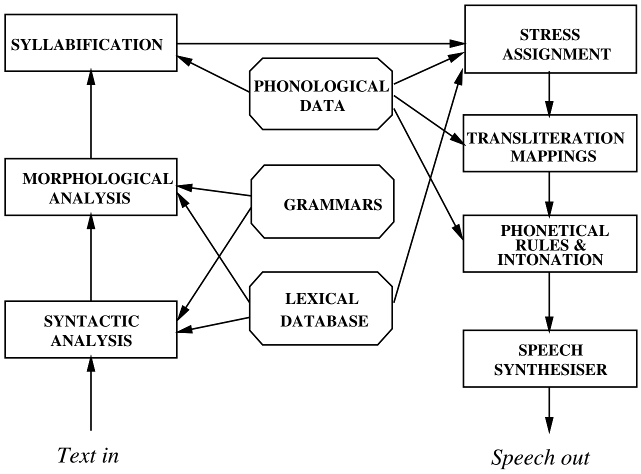


Figure 1: *Modules in GRAFON, a linguistic knowledge-based grapheme-to-phoneme conversion system.*

A problem with this approach is that the knowledge needed is highly language-dependent and requires a significant amount of linguistic engineering. We argue that using data-oriented learning techniques on a corpus of transcribed words (information which is readily available in many machine-readable dictionaries), the same and even better performance can be achieved, without explicit modeling of linguistic knowledge. The advantages of such an approach are that the technique is reusable for different sets of data (e.g. different languages or sublanguages), and that it is automatic (no explicit linguistic engineering is needed to handcraft the rules and knowledge structures necessary for implementing the target mapping).

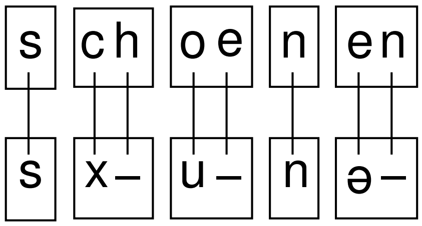
In this paper we present two instances of this approach in the domain of Grapheme-to-Phoneme conversion. A first model implements a variant of *instance-based learning*, in which a similarity metric (weighed by using a metric based on information entropy) and a database of prototypical exemplars are used to predict new mappings. In a second model, grapheme-to-phoneme mappings are looked up in a compressed text-to-speech lexicon (*table lookup*) enriched with default mappings. The most surprising result of our research is that the simplest method (based on tables and defaults) yields the best generalisation results, suggesting that previous knowledge-based approaches to the problem were overkill. For the case of Dutch, we make a comparison of performance and accuracy of these approaches to a connectionist (backpropagation) approach and to a state-of-the-art linguistic knowledge-based approach. To prove reusability of the method, we show how our approach can also be used for automatically generating French and English phonemisation modules.

2 Data-Oriented Text-to-speech Conversion

The algorithms we have applied in our research are *similarity-based* and *data-oriented*. The phonemisation problem is interpreted as a classification task. Given a target grapheme and its context, the corresponding phoneme should be predicted. The algorithms we used to learn this task are *supervised* and *data-intensive* in the sense that a large number of examples is provided of input representations with their correct category (in this case a phonetic transcription). Within a supervised, similarity-based approach, the degree in which abstractions are extracted from the examples may be different, as may be the time when abstractions are created: during training in *aggressive abstraction*, during performance in *lazy learning*. For grapheme-to-phoneme conversion, we claim a data-intensive, lazy learning approach is appropriate to capture the intricate interactions between regularities, subregularities, and exceptions that characterise the domain.

2.1 Training and Test Set Encoding

Training and test set were randomly selected from a Dutch text-to-speech vocabulary data base. From the 70,000 word dataset, 20,000 were randomly selected and randomly divided into 18,500 training words and 1,500 test words. In both sets, each graphemic word is accompanied by its pronunciation in the form of a string of phonemes. In cases where phonemes correspond to grapheme clusters (i.e. there is an alignment problem of grapheme strings with their corresponding phoneme strings), as is the case in, e.g., <*schoenen*> (shoes) /sχunə/, one grapheme of that cluster is algorithmically mapped to the phoneme, and the remaining graphemes are mapped to *phonetic nulls*, represented by hyphens. In the example of <*schoenen*>, this phonetic null insertion results in the following alignment:



To provide a learning system with sufficient information about the phonemisation task, context information must be added. In the models described below, this is done by using graphemic *windows* (compare [Sejnowski and Rosenberg, 1987]), i.e. fixed-length parts of words in which one grapheme is mapped to a phoneme; the other graphemes serve as context. For example, using a window with one left context grapheme and two right context graphemes (from here on written as '1-1-2'), the application of this window on the word <*boek*> (book), pronounced as /buk/, would result in the four pattern-category pairs of Table 1.

This approach implies that dependencies stretching longer than the length of the graphemic window cannot be learned.

2.2 Instance-Based Learning

As an example of a lazy learning approach, we experimented with Instance-Based Learning (IBL, [Aha *et al.*, 1991]). IBL is a framework and methodology for incremental supervised machine learning. The distinguishing feature of IBL is the fact that no explicit abstractions are constructed on the basis of the training examples during the training phase. A selection of the training items themselves is used to classify new inputs. IBL shares with Memory-Based

Pattern nr.	Left context	Focus position	Right context	Target phoneme
1	_	b	o e	b
2	b	o	e k	u
3	o	e	k _	-
4	e	k	_ _	k

Table 1: *Example of the application of the 1-1-2 encoding on the word < boek > (book). Underscores represent spaces, a hyphen represents a phonetic null.*

Reasoning (MBR, [Stanfill and Waltz, 1986]) and Case-Based Reasoning (CBR, [Riesbeck and Schank, 1989]) the hypothesis that much of intelligent behaviour is based on the immediate use of stored episodes of earlier experience rather than on the use of explicitly constructed abstractions extracted from this experience (e.g. in the form of rules or decision trees). In the present context of learning linguistic mappings, the hypothesis would be that much of language behaviour is based on this type of memory-based processing rather than on rule-based processing. In linguistics, a similar emphasis on analogy to stored examples instead of explicit but inaccessible rules, is present in the work of a.o. [Derwing and Skousen, 1989]. IBL is inspired to some extent on psychological research on exemplar-based categorisation (as opposed to classical and probabilistic categorisation, [Smith and Medin, 1981]). Finally, as far as algorithms are concerned, IBL finds its inspiration in statistical pattern recognition, especially the rich research tradition on the nearest-neighbour decision rule (see e.g. [Devijver and Kittler, 1982], for an overview).

2.2.1 Basic Algorithm and Extensions

The main datastructure in our version of IBL is the *exemplar*, a memory structure representing about each pattern the following information: (i) Its distribution over the different categories (training patterns may be ambiguous between different categories, so the memory structure should keep information about how many times each category was assigned to a particular pattern). (ii) Its category. This is simply the category with highest frequency in the distribution of a pattern, or a random selection to break a tie. (iii) Other bookkeeping information (performance data, frequency of pattern in training set, etc.)

Training. For each training pattern, it is checked whether an exemplar for it is already present in memory. If this is the case, the frequency of its category is incremented in the distribution field of the corresponding memory structure. If the new training item has not yet been stored in memory, a new memory structure is created. In learning linguistic mappings (a noisy domain), learning in IBL often is helped by forgetting poorly performing or unrepresentative training items. In this research a simple technique was used to prune memory: each new training item is first classified using the memory structures already present. If it is categorised correctly, it is skipped. We have experimented also with more elaborate storage saving techniques (based on prototypicality and performance of training patterns), but the results are preliminary and will not be reported here.

Testing. If the test pattern is present in memory, the category with the highest frequency associated with it is used. 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) category of the highest ranking exemplar is then predicted as category of the test pattern. When using a Euclidean distance metric (geometrical distance between two patterns in pattern space), all features are interpreted as being equally important. But this is of course not necessarily the case. We extended the basic IBL algorithm proposed by [Aha *et al.*, 1991] with a technique for assigning a different importance to different features. Our approach to the problem of weighing the relative importance of features is based on the concept of *Information Gain* (IG), also used in learning inductive decision trees, [Quinlan, 1986], and first introduced (as far as we know) in IBL in [Daelemans and Van den Bosch, 1992] in the context of a syllable segmentation task. The idea is to interpret the training set as an information source capable of generating a number of messages (the different categories) with a certain probability. The information entropy of such an information source can be compared in turn for each feature to the average information entropy of the information source when the value of that feature is known. The difference is the IG value for that feature. The (normalised) IG value is used as a weight for that feature during similarity matching. Figure 2 shows the pattern of information-gain values for the different positions in the 2-1-3 grapheme window. Unsurprisingly, the target grapheme is most informative, and context features become less informative the further they are removed from the target. We also found that right context is more informative than left context (compare [Weijters, 1991]).

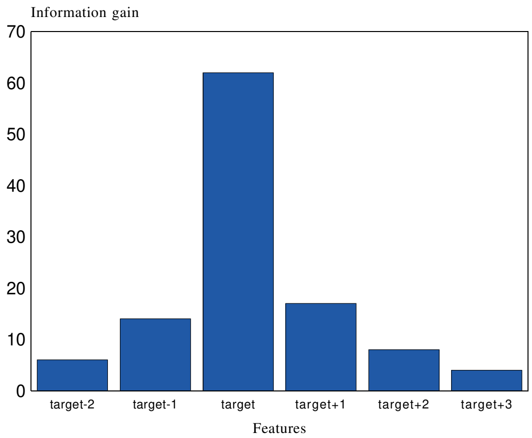


Figure 2: *Information gain value for each position in the 2-1-3 grapheme window.*

2.3 Table Lookup with Defaults

Our table lookup model can be seen as a link between straightforward lexical lookup and similarity-based reasoning. Lexical lookup of word-pronunciation pairs has various disadvantages, an important one being that this approach only works for the words that are stored in the lexicon and not for new words. Without the possibility of manipulating graphemic strings smaller than whole words, there is no way that lexical lookup can provide generalisations on the basis of which new words can be transliterated.

The table lookup model presented here takes as its training set a text-to-speech lexicon, but solves the problems of lacking generalisation power and efficiency by *compressing* it into a text-to-speech *lookup table*. The main strategy behind the model is to dynamically determine which left and right contexts are minimally sufficient to be able to map a grapheme to the correct phoneme with absolute certainty[1]. The context needed to disambiguate a grapheme-to-phoneme mapping can be of very different width. Extreme examples in Dutch are on the one hand the *c-cedille*, present in a small number of loan words (e.g., <*reçu*>), always pronounced as /s/ regardless of left or right context, and on the other hand the <*e*>, which can map to various phonemes (e.g., /ə/, /ɛ/, /e/) in various contexts. For example, the disambiguation of the pronunciation of the final <*e*> in words ending with <*-ster*> (either *star* or female profession suffix) sometimes involves taking into account large left contexts, as in the examples <*venster*> (window) and <*dienster*> (servant), in which the final <*e*> is pronounced /ə/, versus <*morgenster*> (morning star), in which the final <*e*> is pronounced /ɛ/. To disambiguate between these three cases, it is necessary to go back five positions in these words to find the first grapheme which the words do not have in common.

Table Construction. The algorithm starts by searching for all unambiguous one-to-one grapheme-phoneme mappings, and storing these mappings (patterns) in the *lookup table*, more specifically in the 0-1-0 *subtable*. The few unambiguous 0-1-0 patterns in our training set include the <*ç*> - /s/ case mentioned earlier. The next step of the algorithm is to extend the width of the graphemic window by one character. We chose to start by extending the window on the right (i.e., a 0-1-1 window), because, as also reflected earlier in the Information Gain metric used in the IBL model, right context appears to contain slightly more valuable information than the equivalent left context[2]. The algorithm then searches for all certain 0-1-1 patterns to store in the 0-1-1 subtable. Compression is achieved because extensions of unambiguous patterns in the 0-1-0 subtable do not have to be stored in the 0-1-1 subtable. This procedure of extending the window and storing all certain patterns that have not been stored earlier is then repeated (extending 0-1-1 to 1-1-1, then to 1-1-2, etc.), and stops when the whole training corpus is compressed in the lookup table, and all grapheme-phoneme mappings in the corpus are supplied with sufficient left and right contexts. The model evaluated below is calculated up to the 5-1-5 window. At that point, the lookup table covers 99.5% of all grapheme-phoneme mappings in the training set. As a measure of the amount of compression, in number of bytes, the size of the set of linked tables (including the default table discussed below) is 5.8% of the size of the part of the lexicon used as training set[3].

Figure 3 displays the magnitudes of the subtables. It can clearly be seen that most ambiguity is resolved with relatively small contexts. The majority of the ambiguity in the training set is already resolved at the 2-1-2 subtable, after which further extension of window width gradually decreases the number of stored patterns (i.e., resolved ambiguities).

[1]Here, absolute certainty of a grapheme-phoneme correspondence does only express the fact that that correspondence is unambiguous in the training set of the model.

[2]The fact that Information Gain reflects this asymmetry, led us to a new, more generic and domain-independent, conceptualisation and implementation of the Table Lookup method, in which context features are ordered according to their information gain, and patterns are stored in a single *trie* instead of in separate tables.

[3]With considerable further compression when using a trie representation.

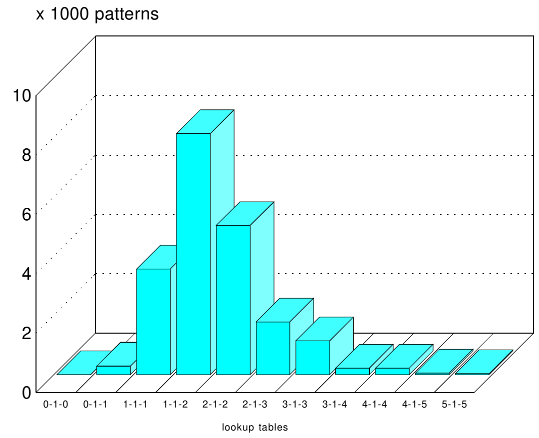


Figure 3: *Table magnitudes of lookup subtables.*

Retrieval. The pronunciation of a word can be retrieved by taking each grapheme of that word separately, and searching in the lookup table for a matching graphemic pattern. First, the grapheme is looked up in the 0-1-0 subtable. If it does not match with any graphemic pattern stored in that table, the single grapheme pattern is extended to a 0-1-1 pattern. This procedure is then repeated until a matching pattern with a minimal context is found, returning a 'certain' grapheme-phoneme mapping. After all graphemes have been processed this way, the phonemic mappings are concatenated to form the pronunciation of the word.

An example of retrieving the pronunciation of a word by table lookup is given in Table 2. As this example illustrates, the contexts needed for disambiguating between output categories are generally very small.

left context	focus grapheme	right context	target phoneme
_	a	a	a
a	a	nb	-
a	n	b	m
n	b	i	b
b	i	ed	i
i	e	di	-
e	d	i	d
d	i	ng	ɪ
i	n	g_	ŋ
n	g	_	-

Table 2: *Example of the correct retrieval of the pronunciation of* < *aanbieding* > *(offer). Each row contains an unambiguous pattern with minimal context found by the lookup algorithm. Underscores represent spaces.*

In case of unseen test words that contain grapheme patterns not present in the training set, the lookup algorithm will not be able to retrieve that specific mapping. This problem is handled in our model by adding to the lookup table a second table which contains all occurring graphemic patterns in the training set of a fixed window width (1-1-1), coupled with their *most frequently occurring* (default) phonemic mapping. Whenever lookup table retrieval fails and a match can be found between the test pattern and a 1-1-1 default pattern, this default table provides a 'best guess' which in many cases still turns out to be correct. To cover for those particular cases where no matching can be found between the test pattern and the 1-1-1 default patterns, a small third default table is added to the model, containing for each grapheme its most frequently occurring phonemic mapping regardless of context (0-1-0), returning a 'final guess'.

It is important to see that *generalisation* in this approach arises from two different mechanisms: (i) the fact that spellings of different words contain identical grapheme substrings, and (ii) the default tables which reflect probabilities of mappings in the training set. More sophisticated reasoning methods can be used instead of the default table: at present we are investigating the consequences of substituting case-based reasoning such as implemented in IBL for the present default tables.

3 Comparison to Alternative Approaches

To get a better insight into the performance of our data-oriented generalisation methods, we compared it to the performance of a 'classical' data-oriented method (backpropagation learning in feed-forward nets), and to the linguistic knowledge-based approach.

3.1 An Alternative Data-Oriented Approach: Connectionism

In IBL, explicit use is made of similarity-based reasoning. A similarity metric is used to compare items, and the items most similar to a test item are taken as a basis for making a decision about the category of the latter. Backpropagation learning in feedforward connectionist networks (BP), too, uses similarity (or analogy), but more implicitly. Again, an input pattern activates an output pattern which is similar to the activation pattern of those items that are similar to the new item. Complexity is added by the fact that an intermediate hidden layer of units "redefines" similarity by extracting features from the activation patterns of the input layer. (In our version of IBL, the information gain metric is used to achieve a similar result).

Automatic learning of grapheme-phoneme conversion of English (NETtalk, [Sejnowski and Rosenberg, 1987]) has been acclaimed as a success story for BP. The approach was replicated for Dutch in NetSpraak ([Weijters and Hoppenbrouwers, 1990]). It is therefore appropriate to compare our two data-oriented models to BP. The main reason for experimenting with BP for learning complex linguistic mappings is that BP networks are alleged to be able to extract generalisations and sub-generalisations from their training data, as well as store exceptions to these generalisations. However, there are limitations to BP network capabilities. BP learning is not guaranteed to converge to optimal performance (i.e. it can end up in local minima). A consequence of this is that although a multi-layered network may be able in principle to represent the solution to any mapping problem, this property is not of much help because the designer of such a network is confronted with a large search space of variable network parameters (e.g., size of the hidden layer, learning rate, number of training cycles, connection weight initialisation) which may affect learning and performance of the network considerably, but which cannot be determined by rule. Experimenters can therefore almost never be sure that their results are optimal.

We trained BP networks on grapheme-phoneme conversion using a training set which was restricted, for computational reasons, to all unique 2-1-3 mappings present in the original training set, removing frequency information, but also removing a large portion of redundancy inherent in the original training set. The same test set was used as with IBL and the table lookup model. Graphemes were encoded by assigning each letter a randomly generated 6-bit string; phonemes were encoded locally, i.e., each phoneme was represented by a 46-bit string with only one phoneme-specific bit 'on'. Systematic simulations were used to approach optimal hidden layer size (60 units), learning rate (0.05) and momentum (0.4). With these parameter settings, 5 networks were randomly initialised (weights were assigned random values between -0.5 and 0.5) and trained for 100 epochs[4]. The performance result listed below was calculated by averaging the performance scores of these 5 networks on the test set.

3.1.1 Results

The performance scores on randomly selected, unseen test words show a surprising best score of the table lookup model. Both IBL and the table perform better than BP. Similar results were obtained for different training and test sets, except that for small training sets, table lookup performance dropped below that of IBL, but was still higher than that of BP.

Model	Generalisation Accuracy on Phonemes
BP	91.3
IBL	93.4
TABLE	95.1

3.2 The Linguistic Knowledge-Based Approach

The traditional linguistic knowledge-based approach of grapheme-phoneme conversion has produced various examples of combined rule-based and lexicon-based models, e.g. MITALK ([Allen *et al.*, 1987]) for English, GRAFON-D ([Daelemans, 1988]), and MORPA-CUM-MORPHON ([Nunn and Van Heuven, 1993]) for Dutch. The developers of all of these models shared the assumption that the presence of linguistic (phonotactic, morphological) knowledge is essential for a grapheme-phoneme model to perform at a reasonably high level.

In MORPA-CUM-MORPHON ([Nunn and Van Heuven, 1993]), a state-of-the-art system for Dutch, grapheme-phoneme conversion is done in two steps. First, MORPA ([Heemskerk and Van Heuven, 1993]) decomposes a word into a list of morphemes. These morphemes are looked up in a lexicon. Each morpheme is associated with its category and a phonemic transcription. The phonemic transcriptions of the consecutive morphemes are concatenated to form an underlying phonemic representation of the word. MORPHON then applies a number

[4]All connectionist simulations were run on PlaNet 5.6, a network simulator written by Yoshiro Miyata (Chukyo U., Japan).

of phonological rules to this underlying representation, deriving the surface pronunciation of the word.

In [Nunn and Van Heuven, 1993], test data is referred to with which they evaluated their system. We applied the table-lookup method to the same test data in order to make a comparison possible. The test file consists of 1,971 words from various sources: newspaper text, compounds, and low-frequency words. High-frequency words, acronyms, and proper names were removed from the original data set. As we did not try to solve the stress assignment problem in this experiment, we had to recompute the error given by [Nunn and Van Heuven, 1993] such that stress assignment or syllable structure errors were not taken into account[5].

The table lookup model was reconstructed on the basis of the complete 70,000 word-pronunciation pairs in our Dutch corpus, resulting in a model containing 48,000 patterns (including default tables).

3.2.1 Results

When comparing the phonemisation accuracy of the linguistic knowledge-based approach in MORPA-CUM-MORPHON to the results on the same data by the table method, we see that the table scores significantly higher.

Model	Generalisation Accuracy on Words
TABLE	89.5
MORPA-CUM-MORPHON	85.3

In the knowledge-based approach, errors of morphological analysis (spurious ambiguity or no analysis) account for a considerable amount of incorrect phoneme output (even after removal by [Nunn and Van Heuven, 1993] of proper names and other difficult cases from the test set). A new data-oriented version of MORPA ([Heemskerk, 1993]) assigns a priority ordering to the set of morphological decompositions, based on a probabilistic grammar derived from a corpus of examples of correct decompositions. This new approach raises the overall performance of MORPA-CUM-MORPHON to 88.7%, which remains slightly worse than the table method.

On the basis of an analysis of confusion matrices (misclassifications per grapheme), we find that the same types of errors are made by both systems, mainly on vowels (especially on the transcription of grapheme <*e*>), but less frequently by the table method. E.g. an intended /ə/ was assigned category /ɛ/ 112 times by MORPA-CUM-MORPHON, and only 23 times by the table method. Another difference is that while confusions by the table method are symmetric, confusions in MORPA-CUM-MORPHON seem to be directed (e.g. an intended /ə/ is often misclassified as /ɛ/, but almost never the other way round).

[5]In a different set of experiments, we successfully applied the IBL approach and two other data-oriented algorithms, analogical modeling and backprop, to the stress assignment problem (see [Gillis *et al.*, 1992], [Daelemans *et al.*, 1993], but we have not yet tried to combine the two tasks.

4 Discussion

4.1 Related Work

As mentioned earlier, Instance-Based Learning is a form of case-based reasoning: a set of exemplars (cases) and a similarity metric are used to make decisions about unseen cases. Earlier work on the application of Memory-Based Reasoning ([Stanfill and Waltz, 1986; Stanfill, 1987]) (another form of case-based reasoning) to the phonemisation problem using the NetTalk data (MBRTalk), showed a better performance than NetTalk itself ([Sejnowski and Rosenberg, 1987]), however at the cost of an expensive, domain-dependent computational measure of dissimilarity that seems to be computationally feasible only when working on a massive parallel computer like the Connection Machine. The Information Gain metric we use in our version of IBL is domain-independent and can be efficiently computed. Another case-based system (or rather a hybrid combination of case-based reasoning and relaxation in a localist interactive activation network) is PRO ([Lehnert, 1987]). The reported performance of this system is not very convincing, however, neither is the need for a combination of connectionist and case-based techniques apparent. Dietterich and Bakiri ([1991]) systematically compared the performance of ID3 ([Quinlan, 1986]) and BP on the NetTalk data. Their conclusion is that BP consistently outperforms ID3 because the former captures statistical information that the latter does not. However, they demonstrate that ID3 can be extended to capture this statistical information. Dietterich and Bakiri suggest that there is still substantial room for improvement in learning methods for text-to-speech mapping, and it is indeed the case that our approach significantly outperforms BP.

The application of compression techniques like our table method to the phonemisation problem has not yet been reported on as such in the literature. In [Golding and Rosenbloom, 1991], the interaction of rule-based reasoning and case-based reasoning in the task of pronouncing surnames is studied. It is claimed that a hybrid approach is preferable, in which the output of the rules is used *unless* a compelling analogy exist in the case-base. If a compelling analogy is found, it overrides the rule output. In this approach, the (hand-crafted) rules are interpreted as implementing the defaults, and the cases the *pockets of exceptions*. Our table-method works along a different dimension: both default mappings and pockets of exceptions are represented in both the table and the default mapping (which as we suggested earlier, we have replaced in the current version by case-based reasoning). Certain mappings are present in the table (which can be interpreted as an automatically acquired rule set), and uncertain cases are handled by the case-base (or default mapping). Future research should make clearer how these two uses of case-based reasoning are related and whether the strengths of both can be combined.

4.2 Multi-lingual Grapheme-to-Phoneme Conversion

To test the reusability of our algorithms to grapheme-to-phoneme conversion in different languages, we applied the table lookup approach to English and French data. For English, we used the benchmark NetTalk corpus as used in [Sejnowski and Rosenberg, 1987]. For French, we extracted word-pronunciation pairs from Brulex, a lexical data base for French ([Content *et al.*, 1990]). Roughly similar models resulted, proving reusability of technique.

In order to be able to compare these models to the Dutch model described in this paper, we selected for each language a comparable data base containing 20,000 word-pronunciation pairs, and divided these data sets in test sets of 1,500 pairs and training sets of 18,500 pairs. Figure 4 displays the resulting lookup tables, including the Dutch lookup table displayed earlier in Figure 3.

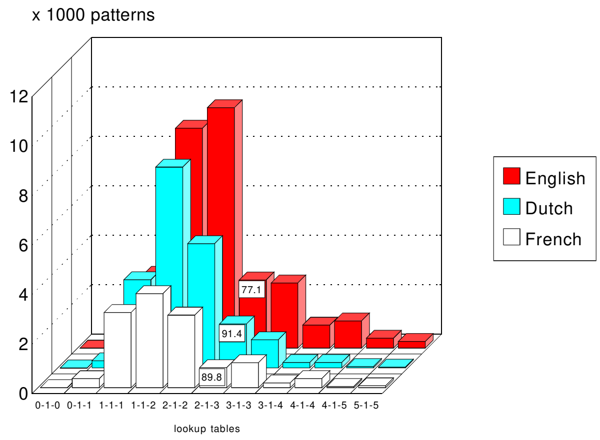


Figure 4: *Table magnitudes of subtables of English, Dutch and French models. For each language, performance accuracy up to the 2-1-3 subtable is displayed on the 2-1-3 subtable bar.*

Given the constant lookup table construction procedure and the similar data sets, two interesting differences between the three models emerge. Firstly, as can be seen clearly from Figure 4, there are significant differences between the magnitudes of the models. After expansion to 5-1-5 width, the French model contains 18,000 patterns, the Dutch 27,000 and the English 35,000, reflecting differences in *deepness* of orthography between the three languages.

Secondly, as the labels on the 2-1-3 subtable bars in Figure 4 indicate, the performance accuracy of the English model lags behind that of the Dutch and French model. Final performance accuracy on the test set, with the inclusion of default tables, is 98.2% for the French model, 97.0% for the Dutch model and 90.1% for the English model, again reflecting a significant difference as regards deepness of orthography between English on the one hand and Dutch and French on the other hand.

5 Conclusion

In computational linguistics, one of the commonsense beliefs is that the performance of a system solving a linguistic problem improves with the amount of linguistic knowledge and sophistication incorporated into it. We have shown that at least for one linguistic task, this is not the case. The linguistically least informed method (compression of a training set into a table, complemented with a rudimentary form of probabilistic reasoning) performed better on unseen input than a linguistically sophisticated, state-of-the-art knowledge-based system. We have reason to believe that this also applies to other linguistic categorisation problems where superficial input features and local context solve most ambiguities (we have positive results on stress assignment, [Gillis *et al.*, 1992; Daelemans *et al.*, 1993], and part of speech tagging).

The data-oriented algorithms described are simple and domain-independent, and introduce a new kind of *reusability* into computational linguistics: reusability of the acquisition method (on different data sets) rather than reusability of (hand-coded) knowledge in different applications or formalisms. The former type of reusability seems to be easier to achieve than the latter.

Acknowledgements

Thanks to Henk Kempff (ITK), Alain Content (ULB), and Terrence Sejnowski (UCSD) for making available for research purposes the Dutch, French and English data we used. We are also grateful to Josee Heemskerk, Anneke Nunn, Gert Durieux, Steven Gillis, Ton Weijters and participants of the CLIN'92-meeting in Tilburg for comments and ideas.

References

[Aha *et al.*, 1991] D. Aha, D. Kibler and M. Albert. Instance-Based Learning Algorithms. *Machine Learning*, 6, 37-66, 1991.

[Allen *et al.*, 1987] J. Allen, S. Hunnicutt, and D. Klatt. *From text to speech: the MITalk system.* Cambridge, UK: Cambridge University Press.

[Content *et al.*, 1990] A. Content, P. Mousty, and M. Radeau. Brulex: une base de données lexicales informatisée pour le francais écrit et parlé. *L'Année Psychologique*, 90, 551-566, 1990.

[Daelemans, 1988] W. Daelemans. GRAFON: A Grapheme-to-phoneme Conversion System for Dutch. In *Proceedings Twelfth International Conference on Computational Linguistics* (COLING-88), Budapest, 133-138, 1988.

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

[Daelemans *et al.*, 1993] W. Daelemans, A. van den Bosch, S. Gillis and G. Durieux. A data-driven approach to stress acquisition. In *Proceedings of the ECML workshop on ML techniques for Text Analysis*, Vienna, 1993.

[Derwing and Skousen, 1989] B. L. Derwing and R. Skousen. Real time morphology: symbolic rules or analogical networks. *Berkeley Linguistic Society*, 15: 48-62, 1989.

[Devijver and Kittler, 1982] P. A. Devijver and J. Kittler. *Pattern Recognition. A Statistical Approach.* London: Prentice-Hall, 1982.

[Dietterich and Bakiri, 1991] T. G. Dietterich and G. Bakiri. Error-correcting output codes: a general method for improving multiclass inductive learning programs. *Proceedings AAAI-91*, Menlo Park, CA, 572-577, 1991.

[Gillis *et al.*, 1992] S. Gillis, G. Durieux, W. Daelemans and A. van den Bosch. Exploring artificial learning algorithms: learning to stress Dutch simplex words. *Antwerp Papers in Linguistics*, 71, 1992.

[Golding and Rosenbloom, 1991] A. R. Golding and P. S. Rosenbloom. Improving rule-based systems through Case-Based Reasoning. *Proceedings AAAI-91*, Menlo Park, CA, 22-27, 1991.

[Heemskerk and Van Heuven, 1993] J. Heemskerk and V. J. van Heuven. MORPA, a lexicon-based MORphological PArser. In V.J. van Heuven and L.C.W. Pols (Eds.), *Analysis and synthesis of speech; strategic research towards high-quality text-to-speech generation.* Berlin: Mouton de Gruyter, 1993.

[Heemskerk, 1993] J. Heemskerk. A probabilistic context-free grammar for disambiguation in morphological parsing. In *Proceedings EACL-93*, Utrecht, 1993.

[Lehnert, 1987] W. Lehnert. Case-based problem solving with a large knowledge base of learned cases. In *Proceedings AAAI-87*, Seattle, WA, 1987.

[Nunn and Van Heuven, 1993] A. Nunn and V. J. van Heuven. MORPHON, lexicon-based text-to-phoneme conversion and phonological rules. In V.J. van Heuven and L.C.W. Pols (Eds.), *Analysis and synthesis of speech; strategic research towards high-quality text-to-speech generation.* Berlin: Mouton de Gruyter, 1993.

[Quinlan, 1986] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1, 81-106, 1986.

[Riesbeck and Schank, 1989] C. K. Riesbeck and R. S. Schank. *Inside case based reasoning.* Hillsdale, NJ: Lawrence Earlbaum Assoc., 1989.

[Sejnowski and Rosenberg, 1987] T. J. Sejnowski and C. R. Rosenberg. Parallel networks that learn to pronounce English text. *Complex Systems*, 1, 145-168, 1987.

[Smith and Medin, 1981] E. E. Smith and D. L. Medin *Categories and concepts.* Cambridge, MA: Harvard University Press, 1981.

[Stanfill and Waltz, 1986] C. W. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29:12, 1213-1228, 1986.

[Stanfill, 1987] C. W. Stanfill. Memory-based reasoning applied to English pronunciation. *Proceedings AAAI-87*, Seattle, WA, 577-581, 1987.

[Weijters and Hoppenbrouwers, 1990] A. Weijters and G. Hoppenbrouwers. NetSpraak: een neuraal netwerk voor grafeem-foneem-omzetting. *Tabu*, 20:1, 1-25, 1990.

[Weijters, 1991] A. Weijters. Analyse van het patroonherkennend vermogen van NETtalk. In J. Treur (Ed.), *NAIC'91 Proceedings*, 249-260, 1991.