Bootstrapping in Language Resource Generation

Marelie Davel and Etienne Barnard

Human Language Technologies
CSIR icomtek, PO Box 395, Pretoria, 0001

mdavel@csir.co.za, ebarnard@csir.co.za

Abstract

This paper describes a method for improving the efficiency of the language resource development process through bootstrapping: iteratively combining machine learning and human knowledge in a way that minimises the human intervention required during the process. Applied to the development of a 10,000-word pronunciation dictionary, it is shown that the amount of human effort can be decreased to less than a quarter of the effort typically required for the manual development of a pronunciation dictionary, without sacrificing accuracy.

1. Introduction

Many speech processing and natural language processing tasks require the availability of extensive language resources: pronunciation dictionaries, large annotated text corpora, annotated speech corpora or parallel text corpora. The development of these resources involves significant effort, and can be a prohibitively expensive task when speech and language technologies are developed for a new language.

Techniques that allow resources to be developed more quickly and cost-effectively include the cross-language re-use of information and bootstrapping approaches. For example, when acoustic models are developed for a new target language, an automatic speech recognition system can be initialised with models from an acoustically similar source language, and these initial models improved through an iterative process in which audio data in the target language is automatically segmented and used to retrain the models. The potential savings in effort achieved through such a process is aptly demonstrated by Schultz [1].

Bootstrapping approaches are applicable to various language resource development tasks, specifically where an automated mechanism can be defined to convert between various representations of the data considered. In the above example, two representations are utilised: annotated audio data and acoustic models, and the mechanisms to move from one representation to the other, are well defined through the phone-alignment and acoustic modelling tasks respectively.

We apply this general approach to the task of creating a pronunciation dictionary, using word/pronunciation pairs and grapheme-to-phoneme (G2P) rules as alternative representations of the same information. In our approach we focus on simplifying and minimising the human intervention required during the bootstrapping process.

The remainder of the paper is organised as follows: Section 2 provides background on the grapheme-to-phoneme conversion problem. Section 3 describes our bootstrapping approach to the creation of pronunciation dictionaries utilising G2P rules as an intermediate representation. Section 4 provides an overview of experimental results.

2. Grapheme-to-phoneme conversion

An accurate model for letter-to-sound conversion is required for various speech processing tasks, including speech synthesis and large vocabulary speech recognition. Typically modelled through explicit pronunciation dictionaries, the relationship can also be described using various letter-to-sound formalisms, including explicit grapheme-to-phoneme mapping rules [2], neural networks [3], decision trees [4] and instance-based learning [5]. A letter-to-sound conversion mechanism is valuable, not only in the absence of pronunciation dictionaries but also to accommodate speech technology on small devices (with associated memory constraints) or to deal with out-of-vocabulary words in speech synthesis.

The results when applying appropriate versions of the different formalisms mentioned above are comparable, with slight variations in performance under specific conditions. For example, neural networks provide stronger generalisation ability than decision trees, and perform more consistently across mismatched test sets, while decision trees typically outperform neural networks where training and test data are closely matched. [6]. Kohonen's Dynamically Expanding Context (DEC), initially applied by Torkkola to the G2P problem [7], is a popular instance-based learning algorithm that predicts phoneme realisation based solely on grapheme context. Variations of DEC typically perform as well, or better, than similar decision tree approaches.

Grapheme-to-phoneme conversion mechanisms can either be defined on a per-grapheme level, or for a combined 'chunk' of graphemes. In the first case it is typically necessary to align each grapheme to a specific phoneme prior to rule extraction. This can be done manually, or through a forced Viterbi alignment, inserting graphemic or phonemic nulls as required [4]. (See section 3.3.)

2.1. DEC

The bootstrapping system described in this paper utilises a variation of DEC as its rule extraction mechanism. In DEC, each rule specifies a mapping of a single grapheme to a single phoneme for a given left and right graphemic context, i.e is of the form: *(grapheme, context)* $\rightarrow$ *phoneme*. Each word in the training dictionary is aligned with its pronunciation on a per-grapheme basis, as illustrated in Table 1. Rules are extracted by finding the smallest context that provides a unique mapping of grapheme to phoneme. If an $n-$letter context is not sufficient, the context is expanded to either the right or the left. This 'specificity order' influences the performance of the algorithm. Different orderings are illustrated in Table 2 as applied to grapheme *'s'* in the word *'interesting'*. Context 1 is expanded symmetrically on a right-grapheme-first basis, context 2 is expanded symmetrically on a left-grapheme-first basis, and context 3 favours the right context on a 2:1 basis.

Table 1: *Word alignment and rule extraction in DEC*

Alignment examples	r o s e → r O z 0 r o w s → r O 0 z r o o t → r u 0 t
Rule examples	o in context -o → u o in context -se → O o in context o- → 0

Table 2: *Different examples of context expansion order in DEC*

size	context 1	context 2	context 3
0	s	s	s
1	st	es	st
2	est	est	sti
3	esti	rest	esti
4	resti	erest	estin

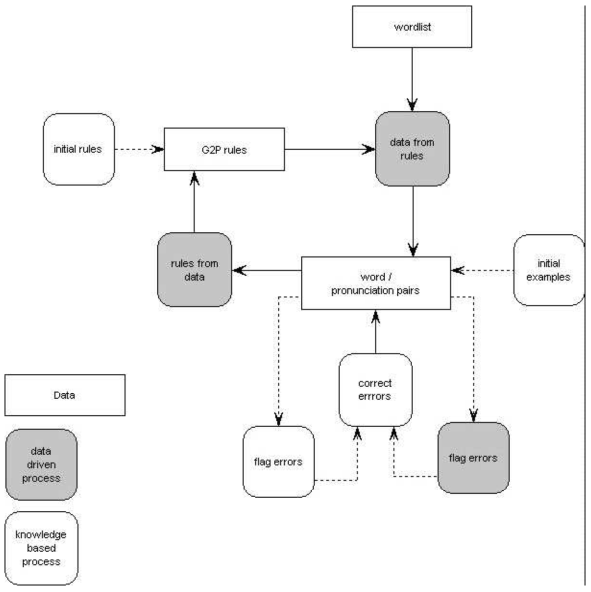


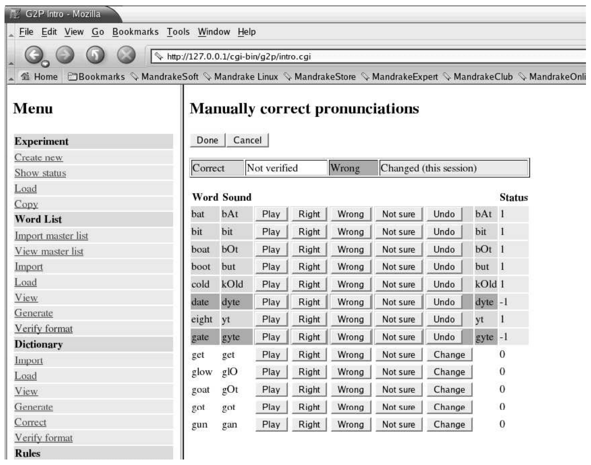
Figure 2: *General bootstrapping system concept*

3. Approach

In this section, we describe our approach to the development of a pronunciation dictionary, both at the process level and with regard to the specific algorithms utilised.

3.1. Process: User perspective

The system is developed to allow a speaker fluent in the target language to develop a pronunciation dictionary without requiring expert linguistic knowledge. The system predicts a pronunciation and presents the human with an audio version of the word: the human acts as a 'verifier' and provides a verdict with regard to the accuracy of the pronunciation (correct, wrong, or uncertain). If the word is wrong, the verifier can either provide the correct pronunciation or flag the phones that are considered wrong. This process is repeated (with increasingly accurate predictions) until a pronunciation dictionary of sufficient size is obtained.


G2P Intro - Mozilla
File Edit View Go Bookmarks Tools Window Help
http://127.0.0.1/cgi-bin/g2p/intro.cgi
Home Bookmarks MandrakeSoft Mandrake Linux MandrakeStore MandrakeExpert MandrakeClub MandrakeOnli

Menu

Experiment
Create new
Show status
Load
Copy
Word List
Import master list
View master list
Import
Load
View
Generate
Verify format
Dictionary
Import
Load
View
Generate
Correct
Verify format
Rules

Manually correct pronunciations

Done Cancel

Correct	Not verified	Wrong	Changed (this session)

Word	Sound							Status
bat	bAt	Play	Right	Wrong	Not sure	Undo	bAt	1
bit	bit	Play	Right	Wrong	Not sure	Undo	bit	1
boat	bOt	Play	Right	Wrong	Not sure	Undo	bOt	1
boot	but	Play	Right	Wrong	Not sure	Undo	but	1
cold	kOld	Play	Right	Wrong	Not sure	Undo	kOld	1
date	dyte	Play	Right	Wrong	Not sure	Undo	dyte	-1
eight	yt	Play	Right	Wrong	Not sure	Undo	yt	1
gate	gyte	Play	Right	Wrong	Not sure	Undo	gyte	-1
get	get	Play	Right	Wrong	Not sure	Change		0
glow	glO	Play	Right	Wrong	Not sure	Change		0
goat	gOt	Play	Right	Wrong	Not sure	Change		0
got	got	Play	Right	Wrong	Not sure	Change		0
gun	gan	Play	Right	Wrong	Not sure	Change		0

Figure 1: *Correcting the predicted pronuncations*

The dictionary correction task as presented to the verifier is shown in Figure 1. This is one task within an experimental environment that allows a user to manipulate and generate the various resources involved (the rule set, word list and pronunciation dictionary) as required. Per experiment, the system logs the history of all activities and archives the intermediary data resources for further analysis.

3.2. Process: System perspective

The general bootstrapping system concept is illustrated in Figure 2. Extracted rules are used to generate additional word/pronunciation pairs, which in turn are used to extract better rules. Errors are flagged when information is presented to the verifier as a list of word/pronunciation pairs. The system is initialised with an initial small rule set or a small transcribed wordlist. If neither is available, the system will predict random pronunciations, which, when corrected, form the basis for further bootstrapping. Currently all verification is done by a human verifier, but data-driven approaches to error identification can be included in the general system, if additional data (such as acoustic information in this example) becomes available.

The overall process consists of the following steps:

- The system analyses its current understanding of the task (number and type of pronunciations from the overall word list that are correct, wrong and uncertain) and generates an optimal list of words to be considered next.
- For each of the words on the above list, the sytem generates a new pronunciation using its current G2P ruleset.
- The system creates a 'sounded' version of each word using the predicted pronunciation and standard IPA sound samples, and records the verifier's response.
- Based on the status of each of the words in the newly verified word/pronunciation list, the system extracts a new G2P ruleset.
- The process is repeated until a sufficient number of correct words are obtained.

3.3. Extracting and applying G2P rules

In order to extract G2P rules, words and pronunciations are first aligned using iterative forced Viterbi alignment. Phonetic nulls are inserted where required (that is, where a single phoneme is produced from more than one grapheme). Graphemic nulls are not used, but graphemic exceptions that can map to more that one phoneme (such as x $\rightarrow$ k s) are replaced with two pseudo-graphemes (as in [8]). The initial probabilities for Viterbi alignment are obtained from words and pronunciations that have equal length.

DEC is used to extract rules of the form *(left context, grapheme, right context)* $\rightarrow$ *phoneme*. Two variations on traditional DEC are implemented:

- Using a sliding window at each context-level, and
- Removing redundant rules.

DEC, as applied by Torkkola [7] expands the context one letter at a time, either favouring the right- or left-hand side explicitly. We use a sliding window that first considers all possible contexts of size *n*, before continuing to consider contexts of size *n*+1, which prevents rules with unnecessarily large contexts from being extracted. In contrast to the DEC context expansion of Table 2, a sliding window applied to grapheme *'s'* in the word *'interesting'* would result in the context ordering indicated in Table 3.

Since multiple rules of the same context size may apply to a single grapheme-to-phoneme mapping (such as *re,s,ti* $\rightarrow$ *s* and *ere,s,t* $\rightarrow$ *s*), contexts that are already served by existing rules are removed to prevent over-specialisation. Because all contexts of each size are considered, the order in which contexts are expanded (for a specific context-level) becomes less significant than in standard DEC. In all experiments, a symmetric right-first expansion scheme is used (as also in Table 3).

Table 3: *Context expansion order in shifted DEC*

order	size	context	order	size	context
1	0	s	2	1	st
3	1	es	4	2	est
5	2	sti	6	2	res
7	3	esti	8	3	rest
9	3	stin	10	3	eres

Generating a new pronunciation is a simple procedure: each grapheme in the word is considered in turn, and the rule describing the largest matching context is used to predict the phoneme to be generated.

3.4. Choosing an optimal word list

An aim of our system is to minimise the number of words a user needs to correct. To achieve this, the system grows its understanding of pronunciations-in-context systematically: ensuring that it achieves certainty on as many contexts of size *n* as possible, before continuing to a context of size *n*+1. For each iteration, it chooses a minimal set of words that covers as many of the contexts in question as possible. The context ordering is chosen to be similar to the one utilised in the G2P extraction mechanism.

4. Experimental results

The approach has been verified in an experimental environment using an existing pronunciation dictionary. The dictionary acts as the verifier, in order to obtain an accurate indication of the process itself, uninfluenced by human error. The pronunciation dictionary used is a hand-crafted dictionary of German words, and consists of 22,000 words and their pronunciations, separated into a 20,000 word training and 2,000 word test set. Results reported are all based on the full test set.

4.1. G2P accuracy

First, the accuracy of the G2P rule extraction mechanism is considered. In all experiments the size of the maxiumum context allowed when extracting rules is not restricted, and the same word training order is used. If DEC is not allowed to grow a context beyond word boundaries and conflicting rules are ignored (*DEC-conflict*) the performance of the algorithm decreases for larger training corpora, especially if rules regarding the context surrounding a grapheme early or late in a word are important. In order to remove this effect, the version of DEC (*DEC-grow*) that was implemented allows a context to grow towards the oposite side if a word boundary is encountered. The shifted window version of DEC (*DEC-win*) outperforms[1] this version of DEC (*DEC-grow*) consistently, as illustrated in Figures 3 and 4.

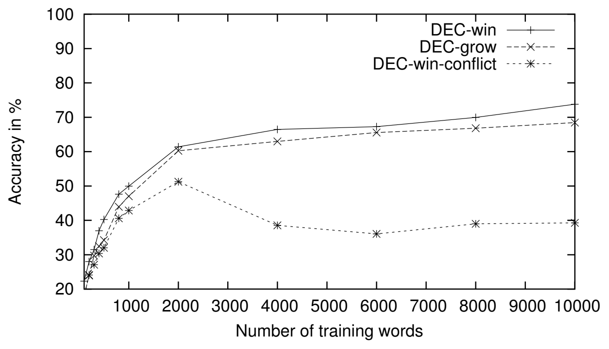


Figure 3: *Word-level accuracy: different DEC variations*

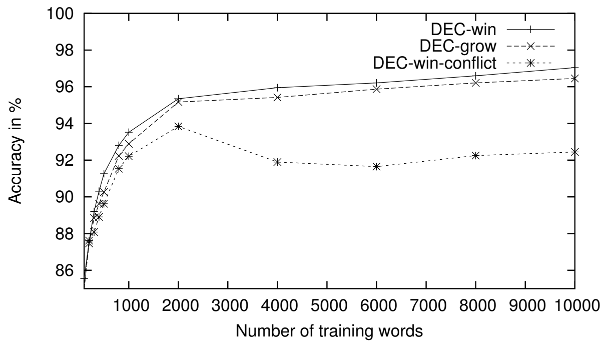


Figure 4: *Grapheme-level accuracy: different DEC variations*

When redundant rules are removed (*DEC-win-opt*) performance is similiar to *DEC-win*, but fewer rules are required to achieve the same accuracy. An analysis of the extracted rules is shown in Table 4. The number of rules of each size (the

[1]Word-level accuracy measures the number of words that are completely correct, while grapheme-level accuracy is measured as the number of correct grapheme mappings minus deletions, divided by the total number of graphemes tested.

size of the context that specifies the rule) is given, as extracted from different sized training dictionaries, using *DEC-win*. For the 10,000-word dictionary, the size of the ruleset obtained via *DEC-win* and *DEC-win-opt* is compared.

Table 4: *Number and size of rules: DEC-win*

Dict size:	10	100	1000	10,000	10,000 (-opt)
Rule size	rules	rules	rules	rules	rules
1	32	32	32	32	32
2	24	90	141	134	128
3	5	85	671	1877	1688
4		7	265	3099	2793
5			23	996	931
6			4	202	190
7			3	73	73
8				37	37
9				17	17
10				7	7
11				1	1
Total	61	214	1139	6475	5877

4.2. Cost-effectiveness

The amount of time required for the various tasks depends on the skills and motivation of the transcriber and verifier. In Table 5 we list rough estimates for relevant times, based on observations in our laboratory. These estimates can be used to predict the effectiveness of the proposed approach compared to that of manual development for pronunciation dictionaries of various sizes. This is illustrated in Figure 5, which shows the predicted amount of human effort required (in hours) as a function of dictionary size when applying *DEC-win-opt*. Additional to the assumptions listed in Table 5, it is assumed that in both approaches each word is verified by a second human verifier.

Table 5: *Assumptions: approximate effort required per task*

Agent	Activity	Approximate effort
Manual transcriber	Transcribe the pronunciation for a given word	90 s
Manual verifier	Verify and possibly correct a transcribed word	60 s
Bootstrap verifier	Listen to and verify a correctly predicted word	15 s
Bootstrap verifier	Listen to and correctly sound an incorrectly predicted word	30 s

The relative benefit of bootstrapping increases for larger pronunciation dictionaries. For a 10,000 word dictionary, the bootstrapping approach requires 23% of the effort of the manual approach (98 hours, compared to 417).

5. Conclusion

A bootstrapping approach to the generation of language resources holds much promise for the accelerated development of Human Language Technologies, especially in the developing world. We have demonstrated that such an approach can

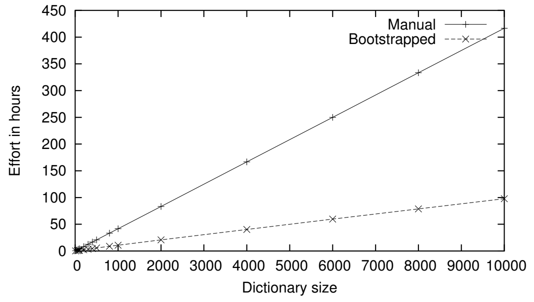


Figure 5: *Effort required: Manual vs Bootstrapped*

significantly reduce the amount of effort required to develop a pronunciation dictionary.

Our process is demonstrated using an existing pronunciation dictionary. We are currently evaluating this same process on the development of a new pronunciation dictionary (for languages where such a resource is not available). This will allow us to measure the performance of these techniques in a realistic context; and to develop credible estimates for the amount of effort required to produce pronunciation dictionaries for all the languages of South Africa.

6. Acknowledgements

This work was supported by CSIR icomtek's *Information Society Technologies Programme*.

7. References

[1] T. Schultz and A. Waibel, "Language-independent and language-adaptive acoustic modeling for speech recognition," *Speech Communication*, vol. 35, pp. 31–51, Aug. 2001.

[2] H. Meng, S. Hunnicutt, S. Seneff, and V. Zue, "Reversible lettter-to-sound generation based on parsing word morphology," *Speech Communication*, vol. 18, pp. 47–63, 1996.

[3] T.J. Sejnowski and C.R. Rosenberg, "Parallel networks that learn to pronounce english text," *Complex systems*, vol. 1, pp. 145–168, 1987.

[4] O. Andersen, R. Kuhn, A. Lazarides, P. Dalsgaard, J. Haas, and E. Noth, "Comparison of two tree-structured approaches for grapheme-to-phoneme conversion.," in *International Conference on Spoken Language Processing*, Philadelphia, 1996, vol. 3, pp. 1700–1703.

[5] Walter Daelemans, Antal van den Bosch, and Jakub Zavrel, "Forgetting exceptions is harmful in language learning," *Machine Learning*, vol. 34, no. 1-3, pp. 11–41, 1999.

[6] J. Hakkinen, J. Suontausta, S. Riis, and K Jensen, "Accessing text-to-phoneme mapping strategies in speaker independent isolated word recognition," *Speech Communication*, vol. 41, pp. 455–467, 2003.

[7] K. Torkkola, "An efficient way to learn english grapheme-to-phoneme rules automatically," in *International Conference on Acoustics and Speech Signal Processing*, Minneapolis, 1993, vol. 2, pp. 199–202.

[8] V. Pagel, K. Lenzo, and A. Black, "Letter to sound rules for accented lexicon compressoin," in *International Conference on Spoken Language Processing*, Sidney, Australia, 1998, vol. 5, pp. 2015–2018.