The Efficient Generation of Pronunciation Dictionaries: Human Factors during Bootstrapping

Marelie Davel and Etienne Barnard

CSIR / University of Pretoria
Pretoria, South Africa
mdavel@csir.co.za, ebarnard@up.ac.za

Abstract

Bootstrapping techniques have significant potential for the efficient generation of linguistic resources such as electronic pronunciation dictionaries. We describe a system and an approach to bootstrapping for the development of such dictionaries, and report on experiments conducted to investigate the efficiency and effectiveness of the system, focusing on the human factors that influence the process. Encouraging results were obtained: even developers with limited linguistic experience can develop accurate pronunciation models in substantially less time than a trained pronunciation expert takes using conventional methods.

1. Introduction

Predicting the pronunciation of a written word is an important component of many speech-processing systems. This can be accomplished with a pronunciation dictionary or a letter-to-sound conversion process (also known as “grapheme-to-phoneme” or “G2P” conversion). The creation of such dictionaries or processes is therefore an important task when developing speech technology for a particular language. Both tasks are highly labour intensive, and can present a significant obstacle to the development of speech technology, especially in the developing world where few electronic resources are available, skilled computational linguists are scarce, and linguistic diversity is high. (India, for example, recognizes 19 official languages and South Africa 11; in countries such as Indonesia and Nigeria, several hundred languages are widely spoken.)

We have therefore proposed an audio-enabled bootstrapping approach to the development of pronunciation dictionaries and/or rule sets[1]. The aim of this approach is to combine machine learning and human intervention during the dictionary creation process in a way that minimizes and simplifies the amount of human effort required. This is achieved by (a) optimizing the speed and accuracy with which the system learns from human input, and (b) minimizing the effort required by the human verifier to identify errors accurately.

Our initial explorations focused on the feasibility of the proposed process, and predicted that significant acceleration could be achieved using bootstrapping. In [2] we describe the techniques implemented to optimise the process from a machine learning perspective. In this paper we describe the techniques implemented to optimise the process from a user perspective, and report on the results achieved. We investigate several practical issues related to the use of bootstrapping in the development of pronunciation rules. In particular, we explore the results obtained when linguistically naive as well as linguistically sophisticated users employ the system to develop pronunciation rules.

2. Background: G2P Bootstrapping

Bootstrapping techniques, including cross-language bootstrapping, have proven useful for the cost-effective development of language resources in new languages[3]. 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. We applied this general approach to the task of creating a pronunciation dictionary, using word/pronunciation pairs and word-to-pronunciation rules as alternative representations of the same information, as described in [1].

Various formalisms have been used to model the relationship between written and spoken words. These include explicit grapheme-to-phoneme mapping rules [4], neural networks [5], decision trees [6] and instance-based learning [7]. The bootstrapping system used in this paper utilises iterative Viterbi alignment to obtain grapheme-to-phoneme mappings, implements a variation of DEC for rule extraction and predicts the next word to verify based on a variety of measures. Details of our implementation can be found in [2].

The bootstrapping system is initialised with a large word list (containing no pronunciation information). The system chooses the next ‘best’ word to consider, predicts a pronunciation for this word and presents a human dictionary developer with an audio version of the predicted pronunciation. The human acts as a ‘verifier’ and provides a verdict with regard to the accuracy of the word-pronunciation pair: whether the pronunciation is *correct* as predicted. The verifier can also indicate that the word itself is *invalid*, *ambiguous* depending on context, or that he or she is *uncertain* about the status. If the word is wrong, the verifier specifies the correct pronunciation by removing, adding or replacing phonemes in the presented pronunciation. A new audio version is generated, for which the verifier can specify a new verdict. At this stage, the learning algorithm updates the word-to-pronunciation model in order to account for the corrected pronunciation. The process is repeated (with increasingly accurate predictions) until a pronunciation dictionary of sufficient size is obtained.

3. Measuring the efficiency of bootstrapping

The system supports the developer to provide input rapidly and accurately. The dictionary developer is presented with each word in turn, and asked to provide a verdict of pronunciation accuracy. Once the word list and phone set have been loaded and the system prepared for the developer, no further expertise is required from the dictionary developer apart from being able to differentiate between correct and incorrect pronunciations. The dictionary developer is presented with two representations of the pronunciation, namely a visual transcription and an audio version. The audio version is created by concatenating pre-recorded samples of each phone (i.e. the word is 'sounded' rather than synthesised). Once the verifier is certain of the accuracy of a specific pronunciation, he or she is encouraged to listen to the audio version of the final pronunciation, and so identify potential errors.

We implemented the bootstrapping approach to run within a Web browser, and measured the efficiency and accuracy of various dictionary developers. Below, we report on one set of experiments, involving three dictionary developers who created pronunciation dictionaries for Afrikaans (a Germanic language with a fairly regular grapheme-to-phoneme relationship). All three developers are first-language Afrikaans speakers. In informal interviews all three were found to employ a broadly similar dialect of "standard" Afrikaans. Two of the developers (whom we will refer to as A and B) have no formal linguistic training, whereas developer C has significant linguistic expertise, and has previous experience in the creation of pronunciation dictionaries.

The experiments are aimed at understanding a number of issues, including the following:

1. Can the bootstrapping approach be used to develop pronunciation dictionaries more quickly than conventional transcription?
2. How important is the linguistic background of the dictionary developer? Is it possible for a first language speaker without any phonetic training to develop an accurate pronunciation dictionary? (As mentioned in Section 1, this is highly significant in the developing world.)
3. How long does it take for a developer to become proficient with the bootstrapping system?
4. What are the practical issues that affect the speed and accuracy of dictionary development using the bootstrapping approach?

To answer these questions, the following protocol was used for all three developers: A brief tutorial on the bootstrapping system, as well as the chosen phonetic representation, was presented by one of the experimenters. A training set of 1000 words was drawn from a corpus of Afrikaans words, and the developers were given the opportunity to familiarize themselves with the system (and the phone set) by developing pronunciation rules for a subset of these words using the bootstrapping system. The process continued until the developers were satisfied that they were comfortable with the software and phone set. A new set of 1000 words was selected, and the developers were asked to produce the most accurate rules they could, by listening to the sounded version produced by the system, correcting it if necessary, and repeating these two steps until satisfied with the pronunciation. Further sets of 1000 words were used to experiment with various other factors, such as the effect of giving developers the option not to use audio assistance. Each set of 1000 words was selected according to the optimal word selection technique described in [2], from an independent 40,000-word subset of the full Afrikaans word list.

During these experiments we measured several relevant variables, including: the time taken to complete each verification; the number of phones changed per word verified; whether the developer chose to use the audio assistance; whether a developer returned to a word to re-correct it at a later stage; and the amount of idle (resting) time between sets of verifications.

4. Experimental results

4.1. Learning to use bootstrapping

To measure a developer's facility in using the bootstrapping software, it is useful to obtain separate measurements of how long it takes (on average) to verify words in which no corrections are made, words where one correction is made, words where two corrections are made, etc. This eliminates the confounding effect of the system becoming more accurate as it learns more rules (thus accelerating apparent developer performance). By this measure, all three developers reached a satisfactory level of performance within approximately 400 words. For example, Fig. 1 depicts how the times for developer C to correct zero through four errors converge to their stable values; similar tendencies were seen for the other developers as well.

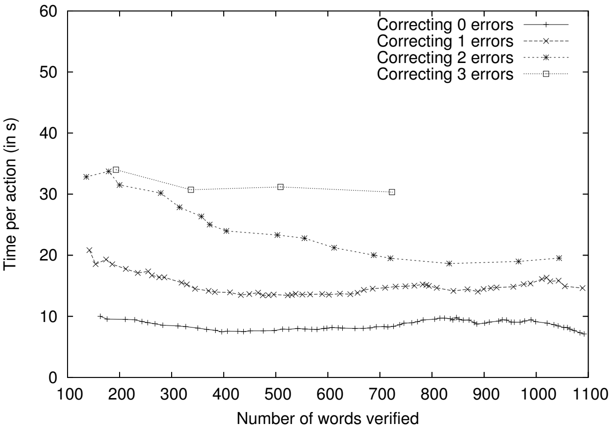

Figure 1: *Average time taken by developer C to verify words requiring zero, one, two or three corrections, as a function of the number of words verified. The averages were computed for blocks of 50 words each.*

This is highly encouraging, since the initial 400 words were completed in less than two hours in every case. Even linguistically untrained users can therefore become proficient at using bootstrapping within this length of time.

4.2. The effect of linguistic sophistication

The performance of developers A and B (who have had no linguistic training) was compared with that of developer C along the dimensions of speed and accuracy. Because there is unavoidable ambiguity in defining "correct" pronunciations (even within a particular dialect), we measured accuracy by manually comparing all cases where any pair of developers chose different transcriptions for a word. In those cases, a transcription was flagged as erroneous if (in the opinion of the authors) it did not represent an accurate transcription of the word.

Table 1: *Estimated transcription accuracies of three developers on a set of 1000 words.*

Developer	Transcription experience	Word accuracy
A	None	83.6%
B	None	98.0%
C	Substantial	99.0%

Table 1 summarizes the accuracies of the three developers, as estimated using this process. Only words marked as "valid" by a developer were included in the evaluation. As expected, developer C was found to be highly accurate. Interestingly, developer B was only slightly less accurate, whereas developer A made significantly more errors than either of the others. During analysis it was revealed that developer A had not adhered to the protocol defined in Section 3: when confident of the accuracy of a pronunciation, developer A had accepted pronunciations without utilising the audio assistance provided by the system. Two conclusions are suggested by these measurements:

- It is possible for a linguistically inexperienced developer to use the bootstrapping system to attain levels of speed and accuracy comparable to those of a highly proficient dictionary developer
- Developers with limited linguistic experience should be required to listen to every transcription, since it is easy to become over-confident about one's ability to read phonetic transcriptions.

4.3. The cost of using audio assistance

Since we found that the developer who did not sound words out made many more errors than those who did, it is important to investigate how much this sub-process delays the process of verification. To this end, we asked developer C to verify an additional set of 200 words, only choosing to sound out those words where she considered it useful. In Fig. 2 the time taken to verify words with various numbers of corrections is compared with the times when the use of audio assistance was compulsory.

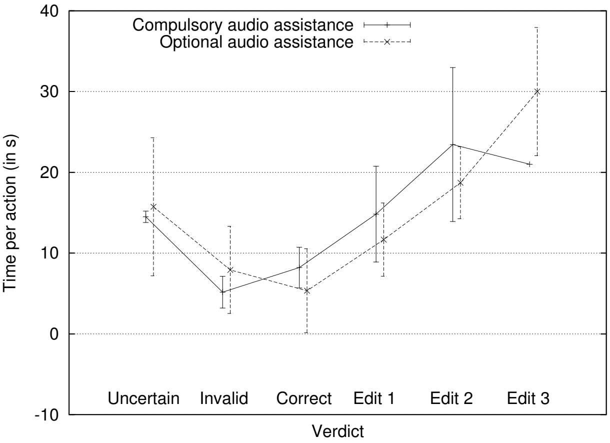

Figure 2: *Average time taken by developer C to verify words, with and without compulsory use of audio assistance.*

We found that this choice did not cause the developer to commit any errors; however, the reduction in verification time was also relatively small (3.6 seconds on average). This confirms the suggestion in Section 4.2 that it is generally better not to make the use of audio assistance optional.

4.4. The efficiency of bootstrapping

As described in Section 3, both human and machine-learning factors contribute to the efficiency of bootstrapping. The increasing likelihood that the system will correctly predict pronunciations as more words are verified is depicted in Fig. 3, which shows the average number of phoneme corrections required as a function of the number of words verified by developer B. The number of corrections decreases steadily as more words are verified, producing an increasingly accurate dictionary and enabling the developer to process subsequent words more rapidly.

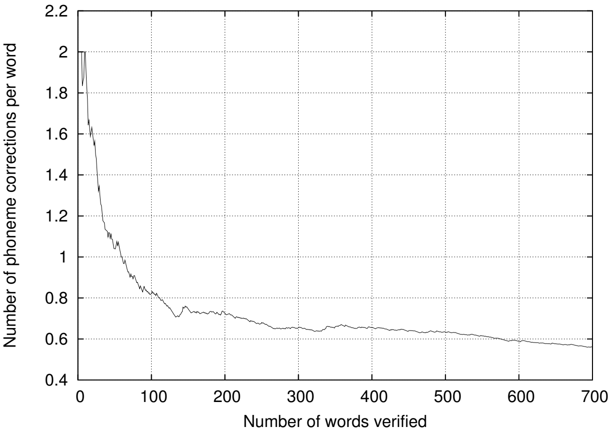

Figure 3: *Expected number of phones that required correction by developer B as a function of the number of words verified.*

The number of phoneme corrections required is the dominant factor in determining verification time. For example, analysis shows that the length of the words to be verified correlates with the verification time if no corrections are required, but not if one correction is required, and that word length is the less important of these two factors. (Word length similarly does not predict verification time if two or more corrections are required.) Developers take comparable durations to perform their verifications, as shown in Fig. 4. A final factor that influences the speed of dictionary development concerns the validity of the initial word lists. In this set of experiments word lists were obtained from Internet text and contained up to 15% invalid words.

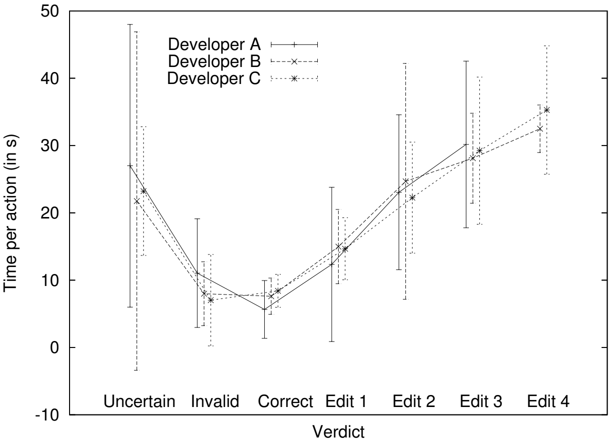

Figure 4: *Average time taken by three developers to verify words requiring different numbers of corrections (or to mark words as invalid or ambiguous/uncertain). The averages were computed for the same set of 1000 words as above.*

We are therefore able to combine the information in Figs. 3 and 4 to derive a model of how long it will take system users such as developers B and C to create pronunciation dictionaries of various sizes. To do this, we fit an exponential curve through the smooth part of the graph in Fig. 3 (i.e., for 100 or more words verified), and estimate a linear model for the expected verification time as a function of the required number of corrections. Fig. 5 shows how machine learning produces slower-than-linear growth in development time, and that a fairly sizeable dictionary can be created in fewer than 20 hours of developer time. The bootstrapping approach is compared to manual verification at 19.2s and 30s per word. (19.2s was the fastest average time observed in our laboratory using a proficient phonetic transcriber, and represents an optimistic time estimate.) Also note that the model of expected development time, which was based on measurements of the time taken by Developer B, predicts Developer C's measurements with reasonable accuracy.

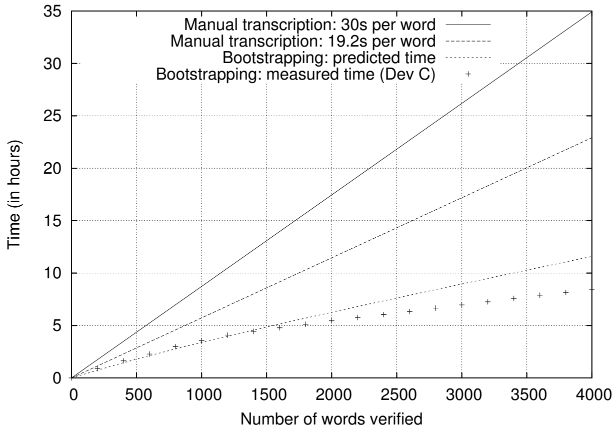

Figure 5: *Expected time (in hours) required to compile an Afrikaans pronunciation dictionary, as a function of dictionary size.*

5. Conclusion

A bootstrapping approach can be used to generate pronunciation dictionaries efficiently. Based on the measurements reported here, we estimate that a 5 000 word pronunciation dictionary for Afrikaans will take approximately 16 hours to compile. Encouragingly, similar estimates are found for an experienced creator of pronunciation dictionaries (with significant linguistic training), and a developer with no prior exposure to formal linguistics. For larger dictionaries, the benefits of bootstrapping increases. The 5000-word dictionary can be used as the basis for predicting a larger dictionary, that will require a steadily decreasing number of phoneme corrections per word.

Our experiments have underlined a number of practical factors that need to be taken into account when developing pronunciation dictionaries using bootstrapping.

- Relatively informal instruction of the developers is sufficient, if they are given the opportunity to learn by using the system.
- The appropriate definition and usage of the phone set requires some care. When a new language is being developed, it is advisable to do this in an iterative fashion: developers develop a small dictionary, and their comments as well as transcriptions are reviewed to determine whether any phones are absent from the set being used, and also to determine what conventions are required to ensure consistency of the dictionary.
- For a linguistically inexperienced dictionary developer, the audio phone set used should ideally match the developer's regional accent.
- When developers have limited linguistic experience, they should be required to listen to every word prior to final acceptance of a transcription.

The experiments reported here involved Afrikaans dictionaries only, but the system was developed to be usable across a range of languages. It seems likely that our findings will generalize to any language which has a fairly regular G2P mapping, and our initial experiments with six languages, including two languages from the Bantu family (isiZulu and Sepedi), supports this conclusion. We will report on findings with those and other languages in future.

Finally, we believe the system can be improved in a number of ways. We have already seen that a separate verification phase, in which the system presents exceptional cases to the developer, can be useful in correcting certain errors, and we plan to develop a definition of "exceptional" that optimizes this benefit. Additional enhancements to the machine learning algorithms are being explored[2] and experiments are currently under way to better understand the implications of different initialisation mechanisms, e.g. when a limited rule set is known prior to dictionary creation, or when a pronunciation dictionary exists in a phonologically similar language.

6. Acknowledgements

This work was supported by the *CSIR Information Society Technologies Centre*, South Africa, as well as the *Local Language Speech Technology Initiative* (LLSTI).

7. References

[1] Marelie Davel and Etienne Barnard, "Bootstrapping for language resource generation," in *Proceedings of the 14th Symposium of the Pattern Recognition Association of South Africa*, South Africa, 2003, pp. 97–100.

[2] Marelie Davel and Etienne Barnard, "The efficient creation of pronunciation dictionaries: Machine learning factors in bootstrapping," in *Proceedings of the ICSLP*, Jeju, Korea, 2004.

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

[4] 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.

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

[6] 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 *Proceedings of the ICSLP*, Philadelphia, 1996, vol. 3, pp. 1700–1703.

[7] 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.