Verifying Pronunciation Dictionaries using Conflict Analysis

Marelie H. Davel and Febe de Wet

Human Language Technologies Research Group,
CSIR Meraka Institute, South Africa
mdavel@csir.co.za, fdewet@csir.co.za

Abstract

We describe a new language-independent technique for automatically identifying errors in an electronic pronunciation dictionary by analyzing the source of conflicting patterns directly. We evaluate the effectiveness of the technique in two ways: we perform a controlled experiment using artificially corrupted data (allowing us to measure precision and recall exactly); and then apply the technique to a real-world pronunciation dictionary, demonstrating its effectiveness in practice. We also introduce a new freely available pronunciation resource (the RCRL Afrikaans Pronunciation Dictionary), the largest such dictionary that currently exists.

Index Terms: pronunciation dictionaries, error detection, quality verification, Default&Refine, grapheme-to-phoneme, g2p

1. Introduction

A comprehensive and accurate pronunciation dictionary remains one of the core components required during the development of many speech technology systems. For literally thousands of languages even basic pronunciation resources do not exist; and for many of the world languages in which large pronunciation dictionaries are widely available, the need remains for the development of such resources tailored to specialized domains, such as geographical place names or medical terms, for example. The current process to develop pronunciation dictionaries requires human intervention, and therefore often introduces human error.

In this paper we describe a new language-independent method for identifying and correcting errors in a pronunciation dictionary, and evaluate the effectiveness of this technique on both synthetic and real-world data. We first provide background to the dictionary verification task (section 2), before describing our specific approach (section 3). We evaluate the effectiveness of our approach using a controlled example (section 4) and demonstrate its performance using real-world data (section 5). The final section (section 6) describes ongoing work.

2. Background

A pronunciation dictionary is both used to provide the most probable pronunciation(s) of words explicitly listed in the dictionary, and to generate grapheme-to-phoneme rules (also referred to as letter-to-sound rules) for unseen words. Such grapheme-to-phoneme (g2p) rules can be extracted in a number of ways, using techniques such as pronunciation-by-analogy [1], decision trees [2], Default&Refine [3], joint sequence models [4] and others.

The process to develop pronunciation dictionaries is labour intensive, even when supported by interactive bootstrapping, as is often the case [5, 6]. Manual annotation of words, or manual verification of annotated words, is often performed by multiple annotators who are required to make subjective judgements with regard to matters that cannot always be captured explicitly in an annotation protocol. Also, being human, small typographical errors and 'convention drift' occur, influencing the accuracy and consistency of the ensuing pronunciation resource.

The extraction of g2p rules provides an immediate avenue for error detection: by cross-validating the pronunciation dictionary (sequentially training g2p rules on one partition of the dictionary and testing on the remainder), errors made by the g2p predictor can be flagged for verification. G2p rules themselves may also be able to identify highly irregular training instances [7] or provide an indication of the likelihood of a specific pronunciation [4, 8] in order to flag possible errors. In related work, g2p accuracy is considered an indicator of dictionary consistency, especially where variants are concerned [9, 10].

3. Approach

In all of the above approaches to error detection, specific samples are flagged for verification and the flagged samples then evaluated individually. However, the conflicting evidence in the dictionary – the specific instances that produce pronunciation patterns conflicting with the flagged samples – are not typically considered. This may hide errors when the flagged pronunciation is actually correct, but could be an indicator of possibly systematic errors occurring in the larger dictionary. Also, when a pronunciation conflict is caused by a difference in convention, neither the flagged pronunciation nor the pronunciation it conflicts with may in itself be incorrect: it is only by viewing the two instances together that it becomes clearer how to resolve the conflict, and potentially also expand the explicit annotation conventions in order to ensure dictionary consistency.

We therefore define a technique to identify not only exceptional instances, but also the specific instances creating the conflict in pronunciation. Our technique utilizes the Default&Refine (D&R) algorithm, a rule extraction algorithm that extracts a set of context-sensitive rules from discrete data and is particularly effective when learning from small rule sets. In addition to an efficient learning curve (learning quickly with little data) the asymptotic accuracy achieved is on par with or outperforms comparative algorithms evaluated against [3].

The D&R algorithm requires the definition of a set of templates and then uses a greedy search to find the most general rule (matching the templates) that describes the training data in question. Patterns that are successfully described by this rule are removed from the data set and the process repeated. Whenever a new rule contradicts examples previously dealt with successfully, these are again added to the training data to be 're-described' by a later rule. Rules are applied according to the

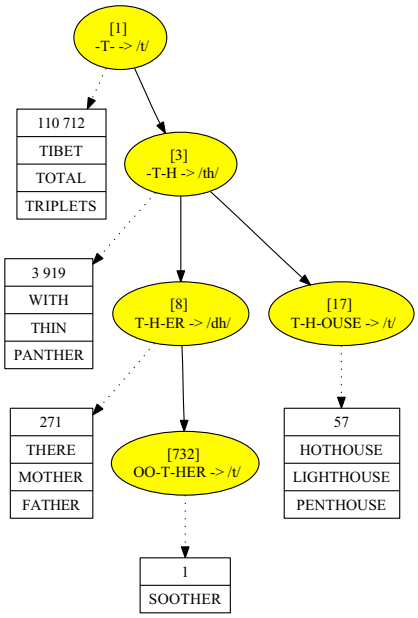

Figure 1: *Small section of a D&R rule set for the grapheme 'T'.*

reverse rule extraction order: the rule extracted first is matched last, and only if no other rule was triggered.

Implicitly, each rule is associated with a number of samples that 'caused' that rule to be generated. In Fig. 1 we depict a small section of a rule set generated from a version of the beep-1.0 dictionary. This approximately 212k-word corpus generates D&R 59 526 rules, 1 129 of which are specifically associated with the grapheme 'T'. Each oval node represents a specific rule, with the rule number provided in brackets. Each associated rectangle indicates the number of instances which 'caused' the rule in question, with three example words displayed. From this rule set example it can be seen that the last rule displayed here indicates a probable error in the pronunciation dictionary. (The word 'soothers' transcribed as */s uh t hh er z/*.)

Note that if a specific number of instances *caused* a rule, there may be more instances that *match* that rule. (Such additional words have already been described by an earlier more general rule: it is incidental that they are now described by an additional more detailed rule as well). We can now define a technique for error detection that consists of the following steps:

1. Extract a set of D&R rules on the full dictionary.
2. Identify all rules generated by instances occurring once, or no more than a set threshold (the *generate-threshold*). These are referred to as the exceptional rules.
3. For each of the exceptional rules, consider how many instances match this rule, and only retain those that display no additional evidence above a second threshold (the *match-threshold*).
4. For each of the remaining instances in this set of rules, list the word(s) that 'caused' that rule. (See above).
5. For each of these words, determine a possible resolution: trace the rule tree to determine the fall-back rule(s) that would have been applied, if the exceptional rule currently being considered did not exist.
6. Provide samples of the words associated with each fall-back rule, and present the conflict to the verifier.

In the above example, the following would be presented to a verifier:

Error?	SOO-T-HERS	→ /t/	/ s uh t hh er z /
Resolution?	SOO-T-HERS	→ /dh/	
Evidence:	SMOO-T-HER	→ /dh/	/ s m uw dh ax r /
	SOO-T-HER	→ /dh/	/ s uw dh ax r /
	...		

In the next section we evaluate this technique using synthetic data, and demonstrate the implications of different threshold settings.

4. Analysis of synthetic data

In order to be able to analyze the effectiveness of the above technique for dictionaries with different types of errors, we first conduct a synthetic experiment using an existing dictionary. We artificially corrupt a fraction of the pronunciations occurring in the Lwazi pronunciation dictionary for Afrikaans (version 1.3) [11] and evaluate the performance of our error detection technique in finding these errors. (Afrikaans is a Germanic language, and one of the official languages of South Africa.)

We introduce two types of corruptions: (1) *Systematic corruptions* reflect the fact that users are prone to making certain transcription errors - for example, in the ARPAbet phone set, *ay* is often used where *ey* is intended. We allow a number of such substitutions, to reflect observed confusions. (2) *Random corruptions* simulate the less systematic errors that also occur in practice; in our simulations, random insertions, substitutions and deletions of phonemes are introduced. We generate 12 corrupted data sets each containing either systematic corruptions, random corruptions or both, where 1%, 2%, 5% and 10% of the words are randomly selected for corruption. (That is, approximately 50, 100, 250 and 500 errors are introduced into different variations of the corpus.) We perform error detection, and evaluate the effectiveness of our technique in finding the known errors introduced during this process.

In Fig. 2 we depict the percentage of errors found using the above technique and both a generate-threshold and match-threshold of 1 (1 instance caused the rule, and no additional words matched the rule). In Fig. 3 we list the number of correct words to verify for every error found (based on the size of

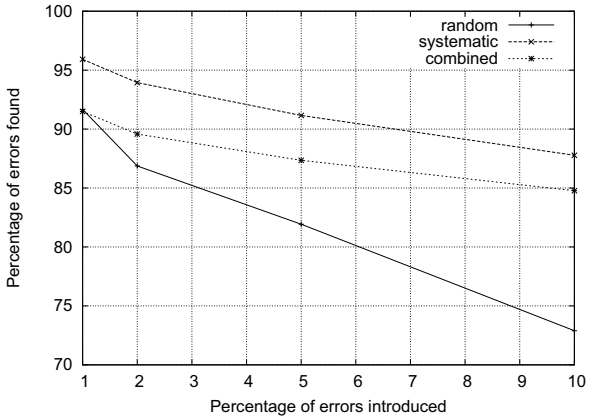

Figure 2: *Percentage of different types of errors found with both a generate-threshold and match-threshold of 1.*

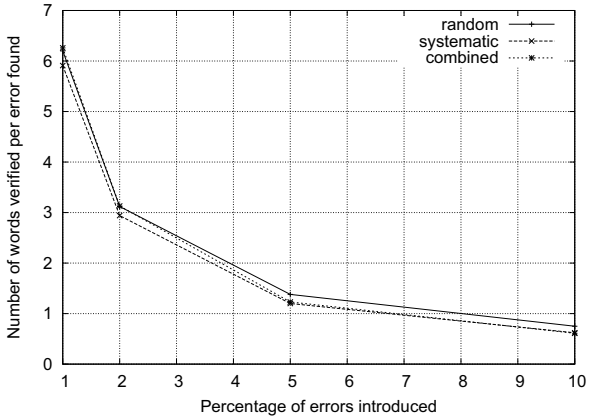

Figure 3: *Number of words verified per error found with both a generate-threshold and match-threshold of 1.*

the set of words flagged as possible errors) for the same settings. As either the generate-threshold or match-threshold is increased, more words are flagged. The additional words flagged do not contribute significantly to the detection of additional errors when the percentage of errors in the dictionary is low. However, when 10% or more of the dictionary contains errors, the looser thresholds do add value. As the match-threshold is increased from 2 to 5 for such less accurate dictionaries, additional errors are found, as listed in Table 1. Note that for dictionaries with less than 5% errors, the technique is less sensitive to threshold settings, with the strictest setting optimal.

Table 1: *Effect of different match-threshold values when a dictionary contains 10% errors, both systematic and random.*

Threshold	Words flagged	Errors found	Precision	Review ratio
1	662	412	84.77	1.61
2	732	420	86.42	1.74
3	795	427	87.86	1.86
4	837	429	88.27	1.95

5. Analysis of real-world data

An initial version of the approximately 24k-word *Resources for Closely Related Languages Afrikaans Pronunciation Dictionary (RCRL APD)* was used in order to validate the error detection process on real-world data. We first provide background with regard to the process that was followed to develop the initial version of the dictionary, and then describe the results of the verification process.

5.1. Dictionary development

The RCRL APD was developed for a project that aims to implement a front-end for an Afrikaans to Dutch speech-to-speech machine translation system. The words were selected from a text corpus containing around 60 million Afrikaans words. This so-called *Taalkommissie Korpus* (TK) was compiled by the Afrikaans language commission as a stratified example of standard, formal Afrikaans in its written form. The corpus comprises many different text types, including newspaper articles, scientific material such as articles, study guides, etc. The sequence in which words were added to the existing dictionary was determined by their frequency of occurrence in the TK corpus, with the most frequent words added first. The frequency count was slightly biased towards the newspaper text in the corpus, because most of the audio data collected for developing the speech front-end corresponded to radio news bulletins.

The RCRL APD was developed by extending the Lwazi pronunciation dictionary for Afrikaans, one of a set of 11 language-specific pronunciation dictionaries developed during project Lwazi [11]. The Lwazi project developed basic but representative speech and language resources for each of South Africa's 11 official languages. As an equal representation of all languages was more important than compiling extensive resources for any specific language, the version of the Lwazi dictionary used (version 1.1) contains only 4,997 entries and a corresponding rule set of 906 D&R rules.

The RCRL APD was developed through a process of interactive bootstrapping, using the Lwazi pronunciation dictionary as source. Two assistants were involved in the process: each assistant was assigned 2 500 new words at a time and in each batch, 200 words were assigned to both assistants. The assistants used the *DictionaryMaker* [12] software tool in order to provide pronunciations for the new words. As DictionaryMaker predicts the most probable pronunciation for each new word (given an underlying set of D&R rules) the assistants were asked to modify the pronunciations suggested by DictionaryMaker rather than to create pronunciations from scratch. After processing each set of 2 500 words, the pronunciations assigned to both the assistants were verified for consistency before proceeding to the next batch of new words. The output of this verification was used to update the transcription protocol and to synchronize the assistants' methodologies. Not all the words that were selected from the TK corpus were incorporated into the dictionary. For example, foreign words were marked for separate handling. (Bootstrapping is most efficient when different dictionaries for different categories of words are developed separately.) The primary goal was to develop a comprehensive and accurate pronunciation dictionary for standard, frequently used words.

5.2. Dictionary verification

Verification was only performed upon completion of the initial dictionary. (Typically verification would be performed more frequently, as the dictionary is developed.) Each verification phase consisted of a number of error detection cycles, each cycle implementing the technique described in section 3. Two additional annotators (not the same assistants used during development) performed verification, working independently. Dictionary verification consisted of three main phases: (1) In the first phase, only those words where the assistants provided identical pronunciations were selected for initial verification. A full dictionary verification was performed, and a new validated subset of the main dictionary created. (2) In the second phase, the subset of words where the assistants disagreed were reviewed and a consensus decision made, also using predictions from the validated subset to guide decisions. Many of these words represented variants, and these were marked as such. (3) In the final phase, the dictionaries from the preceding two phases were combined with all words annotated by individual annotators, and these combined for a final verification.

Table 2 provides a summary of the errors found during the different verification phases. The errors found directly (the exact word was flagged as a possible error) and those found indi-

Table 2: *Errors identified during the various validation phases of the RCRL APD.*

Phase	Num words in dict	New rules flagged	Errors found		
			direct	indirect	total
1 A	16 959	501	97	164	261
1 B	16 959	27	12	11	33
1 C	16 959	8	0	8	8
2 A	17 764	-	-	-	314
2 B	24 201	-	-	-	3
3 A	24 201	889	272	31	303
3 B	24 174	-	-	89	89
3 C	24 174	73	38	37	75
			419	340	

rectly (the error was contained in the conflicting evidence) are reported on separately. Phase 1 consists of three standard error detection cycles (1A to 1C). During phase 2, automated error detection is not performed: direct conflicts between annotators are resolved (2A) and small errors found while formatting the larger dictionary for rule extraction are corrected (2B). A first error detection cycle of the larger dictionary is then performed (3A) and a list of systematic errors compiled based on the results of this cycle. Systematic errors are searched for and corrected first (3B), prior to a final error detection cycle (3C). (Note that the errors found during phase 2 did not make use of the error detection techniques described here, and are therefore not added to either the direct or indirect counts.)

In order to evaluate the effectiveness of the error detection process, we select a random subset of 200 words from the original dictionary (errors included) and verify this manually. We find that of the 18 errors found manually, 16 were correctly identified during the subsequent automated verification. (All errors found during verification were also manually identified.)

5.3. Discussion

From the above analysis, it can be seen that the described technique provides a practical tool for pronunciation dictionary verification. How well does the above method compare to other dictionary verification techniques? This is not an easy question to answer if only considering the precision and recall with which errors are flagged. There are no evaluation corpora specific to this task (no comprehensive dictionaries for which both a (fairly) error-free version, as well as earlier versions containing real-world errors, are available) and what constitutes an error is in some cases clear, and in others highly subjective.

The second aspect of the technique – being able to present verifiers with the specific pronunciations that in some way conflict with the possible error – is not typically done, and from our work this has been shown to be useful in practise. This is especially the case when the flagged pronunciation is not in itself incorrect. For example, in different British English dictionaries, the word 'car' is transcribed either as /k aa/, or /k aa r/, or both options are included as variants. Any one of these conventions has its merits. However, if the conventions are mixed in one dictionary, it can cause problems: (1) the g2p rules become highly complex, (2) unseen words are treated in unexpected ways, (3) it becomes difficult to find other potentially more serious errors, and (4) unnecessary variation is introduced when building speech models, which is especially harmful when developing speech technology in resource-scarce environments.

6. Conclusion

In this paper we describe a new technique for improving the quality and consistency of electronic pronunciation dictionaries. The technique does not only flags specific words for verification, but also presents verifiers with example words that produce pronunciation patterns conflicting with the flagged instances. This assists the verifiers in identifying the cause of an error quickly and is especially useful when the flagged pronunciations are actually correct, but an indicator of possibly systematic errors occurring in the larger dictionary, or when a pronunciation conflict is caused by a difference in convention (rather than an outright error).

Ongoing work includes applying the technique here to an extension of the RCRL dictionary that is more complex in two dimensions: it includes multiple categories of words (each category dealt with individually during verification) and systematic pronunciation variants. The latter is dealt with by combining the technique described here with the use of pseudo-phonemes, as described in [10]. Preliminary results indicate that these techniques generalise well to other languages (including English and Sepedi), but further analysis in this regard is in process.

7. Acknowledgements

This work was supported by the National Research Foundation (NRF) as part of grant FA207041600015 related to *HLT Resources for Closely-Related Languages*.

8. References

[1] Y. Marchand and R. Damper, "A multistrategy approach to improving pronunciation by analogy," *Computational Linguistics*, vol. 26, pp. 195–219, 2000.

[2] A. Black, K. Lenzo, and V. Pagel, "Issues in building general letter to sound rules," in *3rd ESCA Workshop on Speech Synthesis*, Jenolan Caves, Australia, November 1998, pp. 77–80.

[3] M. Davel and E. Barnard, "Pronunciation prediction with Default&Refine," *Computer Speech and Language*, vol. 22, pp. 374–393, 2008.

[4] M. Bisani and H. Ney, "Joint sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.

[5] M. Davel and E. Barnard, "Bootstrapping for language resource generation," in *Proc. PRASA*, Langebaan, South Africa, 2003, pp. 97–100.

[6] S. Maskey, L. Tomokiyo, and A.Black, "Bootstrapping phonetic lexicons for new languages," in *Proc. Interspeech*, Jeju, Korea, October 2004, pp. 69–72.

[7] O. Martirosian and M. Davel, "Error analysis of a public domain pronunciation dictionary," in *Proc. PRASA*, Pietermaritzburg, South Africa, Nov 2007, pp. 13–18.

[8] Y. L. P Vozila, J Adams and R. Thomas, "Grapheme to phoneme conversion and dictionary verification using graphonemes," in *Proc. Eurospeech*, Geneva,Switzerland, 2003, pp. 2469–2472.

[9] M. Wolff, M. Eichner, and R. Hoffmann, "Measuring the quality of pronunciation dictionaries," in *Proc. PMLA*, 2002, pp. 117–122.

[10] M. Davel and E. Barnard, "Developing consistent pronunciation variants," in *Proc. Interspeech*, Pittsburgh, PA, Sept 2006, pp. 1760–1764.

[11] M. Davel and O. Martirosian, "Pronunciation dictionary development in resource-scarce environments," in *Proc. Interspeech*, Brighton, UK, Sept 2009, pp. 2851–2854.

[12] M. Tempest and M.Davel, "DictionaryMaker 2.16 user manual," Sept 2009, http://dictionarymaker.sourceforge.net.