Comparative Objective and Subjective Evaluation of Three Data-Driven Techniques for Proper Name Pronunciation

Tasanawan Soonklang †, Robert I. Damper †, Yannick Marchand ‡

† School of Electronics and Computer Science
University of Southampton, Southampton SO17 1BJ, UK.

‡ Institute for Biodiagnostics (Atlantic)
National Research Council Canada, Neuroimaging Research Laboratory
1796 Summer Street, Suite 3900, Halifax, Nova Scotia, Canada B3H 3A7.

†{ts03r|rid}@ecs.soton.ac.uk, ‡yannick.marchand@nrc-cnrc.gc.ca

Abstract

Automatic pronunciation of unknown words is a hard problem of great importance in speech technology. Proper names constitute an especially difficult class of words to pronounce because of their low frequency of occurrence and variable origin. In this paper, we compare three different data-driven approaches which use a dictionary of (known) proper names to infer pronunciations for unknown names, namely: pronunciation by analogy (PbA), the table look-up method by Weijters, and the 'improved' table look-up method by Daelemans and van den Bosch. Evaluation is both *objective*, in which inferred pronunciations are compared with 'gold standard' dictionary pronunciations, and *subjective*, in which listeners rated synthesised pronunciations using a 5-point scale. Objective evaluation used a leave-one-out technique on 52,911 names in the CMUDICT dictionary. Results show that PbA achieves the best performance at 63.93% names correct. In the subjective evaluation of 529 different pronunciations of 200 names, 12 listeners rated the pronunciations. Non-parametric tests of significance show that the dictionary pronunciations are rated superior to the automatically-inferred pronunciations; PbA is superior to both table look-up methods, and the 'improved' table look-up is superior to Weijters' original method (Walsh test, $p < 0.005$ in all cases.)

1. Introduction

Text-to-speech (TTS) synthesis technology, which has developed enormously over the past decade, enables a voice interface for many commercial applications, especially in telecommunication services. These applications often require the pronunciation of proper names, i.e., the names of people, streets, cities, places, companies, etc. The automatic pronunciation of proper names remains a real challenge for text-to-phoneme conversion, a component part of the pronunciation module in TTS synthesis, because the geographical and language origin of the names can be varied, and the number of distinct names is very large [1, 2].

Numerous studies have proposed various approaches to the problem of proper name pronunciation and have reported the results in different ways. Most studies evaluate the performance objectively in terms of word accuracy, by comparing the pronunciations generated by their models with these in some data set taken as a 'gold standard'. However, the pronunciation should also be acceptable to users of the TTS system, which obviously includes members of the general public with minimal exposure to synthetic speech. Thus, subjective evaluation is also important, particularly for proper names, which can be pronounced differently depending on the linguistic background of the speaker and other cultural factors. Because subjective evaluation is difficult compared to objective evaluation, and cannot deal with large numbers of names, there have been few attempts to do this (but see [3] who have conducted informal subjective evaluation of proper name pronunciations over the internet).

In this paper, we compare three data-driven methods for proper name pronunciation, namely: pronunciation by analogy (PbA), the table look-up method described by Weijters [4], and the table look-up method by Daelemans and van den Bosch [5]. These methods were selected on the basis that, first, we believe PbA to be the best currently-available technique for pronunciation of common words [6] and, second, table look-up is very representative of the competitor data-driven techniques to PbA. Indeed, table look-up can be seen as an alternative implementation of the broad concept of 'analogy'. The comparison involves both objective and subjective performance.

2. Overview of the techniques

Data-driven approaches to letter-to-phoneme conversion generally require the letters of each word in the dictionary to be aligned with the corresponding phonemes, so converting the problem of transduction into one of classification. For the three techniques compared here, the algorithm described in [7] was used for alignment.

2.1. Pronunciation by analogy

The assumption underlying PbA is that the dictionary contains implicit phonological knowledge which can be exploited to generate a pronunciation for an unknown word. An early and influential PbA system was PRONOUNCE [8], and many variants of PbA have since been based on it (e.g., [9, 10, 11, 12, 13]). The particular variant used here is now described.

When an unknown word is presented as an input to the system, so-called full pattern matching (see [12]) between the input letter string and dictionary entries is performed, starting with the initial letter of the input string aligned with the end letter of the dictionary entry. If common letters are found in matching positions in the two strings, their corresponding phonemes (according to the prior alignment) and information about their positions in the input string are used to build a pronunciation lattice, as detailed next. One of the two strings is then shifted relative to the other by one letter and the matching process continued, until the end letter of the input string aligns with the initial letter of the dictionary entry.

The pronunciation lattice is a directed graph that defines possible pronunciations for the input string, built from the matching substring information. A lattice node represents a matched letter, L_i, at some position, i, in the input. The node is labelled with its position i and the phoneme corresponding to L_i in the matched substring, P_{im} say, for the mth matched substring. An arc is labelled with the phonemes intermediate between P_{im} and P_{jm} $(j > i)$ in the phoneme part of the matched substring and the frequency count, increasing by one each time the substring with these phonemes is matched during the search through the lexicon. Arcs are directed from i to j. If the arcs correspond to bigrams, the arcs are labelled only with the frequency. (The string of phonemes intermediate between P_{im} and P_{jm} is empty.) Phonemes P_{im} and P_{jm} label the nodes at each end of the arc, i.e., i and j respectively. Additionally, there is a *Start* node at position 0 and an *End* node at position equal to the length of the input string plus one.

Finally, the decision function identifies the 'best' candidate pronunciation of the input according to some criterion. Possible pronunciations for the input correspond to the string assembled by concatenating the phoneme labels on the nodes or arcs in the order that they are traversed in moving through the lattice from *Start* to *End*. If there is just one candidate corresponding to a unique shortest path, this is selected as the output. If there are tied shortest paths, then the total product (TP) scoring strategy of [11] is used to select the output. The TP score is the sum of the product of the arc frequencies for all shortest paths giving the same pronunciation.

2.2. Table look-up I (TLUI)

This method was proposed by Weijters [4] who argued that his simple look-up procedure is superior to NETtalk, the well-known neural network for letter-to-phoneme conversion [14]. The first step is to create from a training set a table containing n-grams (strings of n letters, n odd), the corresponding phoneme(s) for the middle letter of each n-gram, and their frequencies in the training data. Here, we have used $n = 7$ (heptagrams). One heptagram is produced for each letter of the input, with each letter serving in turn as the centre of the heptagram. To obtain the pronunciation for an input string, we search for the *closest-fit* heptagrams, i.e., those with the highest matched value between the heptagram of the input string and those of the pre-compiled look-up table. The matched values are calculated as in the pseudocode below:

```
Weight[1..7]:={ 1, 4, 16, 64, 16, 4, 1 }
MatchValue := 0
for i := 1 to 7 do
  Begin
  if windowI[i] = windowL[i] then
    MatchValue := MatchValue + Weight[i]
  end if
end for
```

Here, a heptagram in an input is referred to as `windowI`, a heptagram in a look-up table is `windowL`, and i is an index into a heptagram. Weijters actually used a range of different values of n and weight sets, but results did not differ too much for the different choices.

After matching, the phonemes of the closest-fit heptagrams are concatenated to form the pronunciation of the word. If closest-fit heptagrams are tied and correspond to different phonemes, the phoneme that occurs most frequently is chosen. If the frequencies are equal, the first one of the tied phonemes is chosen arbitrarily.

2.3. Table look-up II (TLUII)

Daelemans and van den Bosch [5] describe a method similar to Weijters', but having defaults that are invoked in the case of matching failure so as to improve generalisation ability. During table construction, all unambiguous one-to-one letter-to-phoneme mappings are found and stored in the 0-1-0 subtable. Then, the width of the letter window is expanded on the right by one character, and all unambiguous 0-1-1 patterns found and stored in the 0-1-1 subtable, excluding those patterns already in the 0-1-0 subtable. Then, the window width is expanded on the left by one character and the procedure repeated. The process of expanding the window on right or left and storing all the patterns that have not been stored in the earlier table continues until all patterns in the training set are compressed in the look-up table. In this work, this occurs with a 10-1-10 window. Additionally, two default tables are assembled to provide generalisation ability. The first default table, referred as a *best-guess* table, contains all occurring 1-1-1 patterns and their most frequently occurring phonemic mapping. The second table, referred as a *final-guess* table, contains all letters and their most frequently occurring phonemic mappings.

The conversion algorithm starts by searching for a matching letter pattern for each letter of an input word in the 0-1-0 subtable. Note that, if found, this is guaranteed to be unambiguous. If no pattern is matched, each letter is extended to a 0-1-1 pattern and the 0-1-1 subtable is then searched. This is repeated until a matching pattern with a minimal extension is found and the corresponding letter-phoneme mapping is returned. If no matching can be found, the best-guess table is scanned to return the 'best' mapping. If look-up table retrieval fails again, the default phoneme of that letter from the final-guess table is returned. Finally, all phonemes are concatenated to create the pronunciation of the word.

3. Materials

Test data were developed using a list of proper names (without pronunciations) together with the standard CMUDICT dictionary (version 0.6). The first file can be downloaded from `www.festvox.org/cmunames.lex.gz` and the latter from `www.speech.cs.cmu.edu/cgi-bin/cmudict`. CMUDICT is a pronunciation dictionary containing approximately 100,000 word spellings and their transcriptions, including the most frequent names in the United States (as of some 20 years ago) and their pronunciations [15]. This includes names from a wide variety of origins. The phoneme set for CMUDICT consists of 39 phoneme symbols. The procedure was simply to extract from CMUDICT pronunciations for the names on the first list. Note, however, that some names in this list were not found in CMUDICT. The number of proper names used in this work is 52,911. The listed pronunciations were taken to be the correct 'gold standard' against which to assess the automatically-inferred pronunciations.

For TLUII, no unambiguous 0-1-0 patterns were actually found in CMUDICT. Nonetheless, the general description of the method in section 2.3 above remains valid.

Table 1: *Evaluation of pronunciations of 52,911 proper names by three automatic methods in terms of words correct and phonemes correct.*

	words (%)	phonemes (%)
PbA	63.93	91.03
TLUI	58.55	91.26
TLUII	61.89	92.14

4. Experimental results

In this section, we present the results obtained for objective and subjective evaluation of the three automatic pronunciation approaches as described in Section 2.

4.1. Objective evaluation

The performances of the three methods were evaluated using a leave-one-out strategy, also known as k-fold cross validation. That is, in the case of PbA, each word was removed in turn from the dictionary and a pronunciation derived by analogy with the remaining words. In the case of table look-up, a note was made during table compilation of those heptagrams which were unique to a particular word and that word was stored with the heptagram. Then, when finding a pronunciation for a particular word, heptagrams unique to that word were removed from the look-up table. Frequencies were left unaltered because of the computational difficulty of adjusting them to account for the separate removal of over 50,000 entries. (Contemporary work has shown that 'full' k-fold cross validation is computationally feasible but we nonetheless expect this simplified approach to produce similar results.)

Results were obtained by scoring the automatically-derived pronunciations against the pronunciations in the name list, in terms of both words correct and phonemes correct. We have previously argued that words correct is a more stringent measure of pronunciation accuracy and should be used in strong preference to phonemes correct [6]. As can be seen in Table 1, PbA achieved the highest percentage of words correct. The difference between word accuracy for PbA and for the next best method (TLUII) is enormously significant (binomial test, $z \sim 9.8$, $p \sim 0$). As expected, TLUII achieved enormously better performance than TLUI because of the extension to include default tables. We note that the phonemes correct measures do not align with words correct. TLUII achieves the best phonemes correct score but not the best words correct score. This indicates that phoneme errors are not independently distributed across words, and is another reason for preferring words correct as our measure of effectiveness. Note finally that, for simplicity, we have ignored stress assignment in this objective evaluation.

4.2. Subjective evaluation

To assess the acceptability of the automatically-derived pronunciations from the potential users' point of view, and as a further comparison of the three data-driven methods studied in this work, we employ subjective evaluation by listeners of names synthesised according to the three techniques. As a baseline, listeners also hear pronunciations according to the CMUDICT transcription. To keep the test times tractable, 200 names were randomly selected from CMUDICT and pronunciations obtained using each of the three data-driven methods. In the event that all three methods gave the same (correct) pronunciation, this was discarded and another word selected. At the end of this process, we had 200 names whose pronunciations either differed according to the data-driven method used, or were incorrect (according to CMUDICT), or both. Finally, any duplicate pronunciations were deleted to give a total of 529 different pronunciations ($529 < 200 \times 4$). These were then synthesised by diphone synthesis and played to listeners.

PROPER NAME PERFORMANCE EVALUATION

SUBJECT INFORMATION:

Name .... Age ....

GUIDELINES

This is an evaluation of a speech synthesis system for proper names. Please follow the instructions:

- Listen carefully to the pronunciations of the names on the provided list.
- Rate the pronunciations using the following scales:
    1. Unacceptable, no one would ever pronounce this name like this
    2. Poor, very few people would recognise the name from this pronunciation
    3. Acceptable, some people would recognise this name
    4. Almost correct, most people would recognise this name
    5. Correct, almost everybody would recognise this name

NOTES

- Apostrophes (e.g., in "O'Connell") have been removed in the listed names.
- You will hear one name every 2 seconds.
- There will be a short break (20 seconds) at the bottom of each page.
- You should assess the pronunciations as if they had been produced by a telephone directory assistance service.

Figure 1: *Instructions to subjects for the subjective evaluation.*

Speech output used Festival, a public domain system intended for speech synthesis research available from `http://www.cstr.ed.ac.uk`. To achieve good pronunciations, it was necessary to syllabify the words and add stress. Syllabification was done using the Festival function `lex.syllabify.phstress(PHONELIST)`. To obtain stress patterns, we passed the spelling patterns through Festival's letter-to-sound rules and then manually transferred the stress pattern of the output to the 200 proper names. The voice used was male KAL.

Twelve listeners, 10 male, 2 female, and all native speakers of British English, aged between 23 and 66 years, listened to the synthesised names over headphones in the soundproof room in our laboratory at Southampton. We decided against online internet-based evaluation as used by Font Llitjós and Black [3] because of the lack of experimental control this allows. Subjects were instructed to rate their opinions on the quality of the pronunciations on a five-point scale, according to the mean opinion score (MOS) [16]. Instructions to the subjects are shown in Figure 1. Ratings were recorded by the listeners on a hardcopy list of the name spellings, so that they could match the pronunciations heard with their spellings. The complete experiment took some 30-35 minutes per subject.

Results are listed in Table 2. MOS values generally exceed 3.5 (i.e., tending towards 'Almost correct') indicating that all methods gave reasonable pronunciations. As expected, dictionary pronunciations have a higher MOS than any of the automatically-inferred methods; since the dictionary acts as the

Table 2: *Results of subjective evaluation (mean opinion score, MOS, and its standard deviation, SD) by twelve listeners of pronunciations of 200 proper names.*

	MOS	SD
CMUDICT	3.917	0.376
PbA	3.816	0.406
TLUI	3.636	0.397
TLUII	3.689	0.408

knowledge base for the data-driven methods, there is no basis on which they could systematically improve on CMUDICT. From best to worst in terms of MOS, the data-driven methods are ordered PbA, then TLUII, then TLUI. However, according to a paired t-test (assuming that scores can be treated as interval data and are normally distributed), only the difference between the dictionary pronunciations and TLUI reaches even marginal significance ($t = 1.706$, $p \sim 0.10$, $\text{df} = 12 + 12 - 2 = 22$). It was striking, however, that according to mean scores, all listeners but one consistently rated pronunciations in the order CMUDICT, PbA, TLUII, TLUI. The one exceptional subject reversed the order of TLUI/II. Thus, it seems that the data may not be normally distributed. According to a non-parametric Walsh test [17], which assumes symmetrical (but not normal) interval data, the differences between *all* methods are highly significant ($p < 0.005$). The Walsh test assumes related samples, which is appropriate because the listeners are the same.

5. Conclusions

We have compared three data-driven methods for proper name pronunciation: pronunciation by analogy (PbA), the table look-up method by Weijters (TLUI), and the table look-up method by Daelemans and van den Bosch (TLUII). The best result for objective evaluation was 63.93% words correct, achieved using PbA. This was enormously significantly better that the results of either table look-up method. In the latter case, TLUII was better than TLUI reflecting its superior generalisation ability. In the subjective evaluation, listeners generally rated pronunciations somewhere between 'Acceptable' and 'Almost correct' for all four conditions (dictionary pronunciation plus the three automatic methods). Although parametric paired t-tests did not show significant differences between methods, it was clearly evident in the MOS data that listeners consistently rated dictionary pronunciations best, followed by PbA, followed by TLUII, with TLUI rated worst. A non-parametric Walsh test showed that the differences between *all* methods were significant ($p < 0.005$).

In future work, we plan to add stress to the objective evaluation, test a wider range of methods (perhaps including proprietary rules if we can gain access to a set), to include automatic stress assignment, to improve PbA by using a multi-strategy approach as proposed in [12, 13] and to incorporate automatic syllabification [18].

6. References

[1] T. Vitale, "An algorithm for high accuracy name pronunciation by parametric speech synthesizer," *Computational Linguistics*, vol. 17, no. 3, pp. 257–276, 1991.

[2] M. F. Spiegel, "Proper name pronunciations for speech technology applications," *International Journal of Speech Technology*, vol. 6, no. 4, pp. 419–427, 2003.

[3] A. Font Llitjós and A. W. Black, "Evaluation and collection of proper name pronunciations online," in *Proceedings of 3rd International Conference on Language Resources and Evaluation, LREC2002*, Las Palmas, Spain, 2002, pp. 247–254.

[4] A. Weijters, "A simple look-up procedure superior to NETtalk?" in *Proceedings of International Conference on Artificial Neural Networks (ICANN-91)*, vol. 2, Espoo, Finland, 1991, pp. 1645–1648.

[5] W. Daelemans and A. van den Bosch, "Tabtalk: Reusability in data-oriented grapheme-to-phoneme conversion," in *Proceedings of 3rd European Conference on Speech Communication and Technology, Eurospeech'93*, vol. 2, Berlin, Germany, 1993, pp. 1459–1466.

[6] R. I. Damper, Y. Marchand, M. J. Adamson, and K. Gustafson, "Evaluating the pronunciation component of text-to-speech systems for English: A performance comparison of different approaches," *Computer Speech and Language*, vol. 13, no. 2, pp. 155–176, 1999.

[7] R. I. Damper, Y. Marchand, J.-D. S. Marsters, and A. I. Bazin, "Aligning text and phonemes for speech technology applications using an EM-like algorithm," *International Journal of Speech Technology*, vol. 8, no. 2, pp. 149–162, 2005.

[8] M. J. Dedina and H. C. Nusbaum, "PRONOUNCE: A program for pronunciation by analogy," *Computer Speech and Language*, vol. 5, no. 1, pp. 55–64, 1991.

[9] K. P. H. Sullivan and R. I. Damper, "Novel-word pronunciation: A cross-language study," *Speech Communication*, vol. 13, no. 3–4, pp. 441–452, 1993.

[10] F. Yvon, "Grapheme-to-phoneme conversion using multiple unbounded overlapping chunks," in *Proceedings of Conference on New Methods in Natural Language Processing (NeMLaP-2'96)*, Ankara, Turkey, 1996, pp. 218–228.

[11] R. I. Damper and J. F. G. Eastmond, "Pronunciation by analogy: Impact of implementational choices on performance," *Language and Speech*, vol. 40, no. 1, pp. 1–23, 1997.

[12] Y. Marchand and R. I. Damper, "A multistrategy approach to improving pronunciation by analogy," *Computational Linguistics*, vol. 26, no. 2, pp. 195–219, 2000.

[13] R. I. Damper and Y. Marchand, "Information fusion approaches to the automatic pronunciation of print by analogy," 2005, *Information Fusion*, in press.

[14] T. J. Sejnowski and C. R. Rosenberg, "Parallel networks that learn to pronounce English text," *Complex Systems*, vol. 1, no. 1, pp. 145–168, 1987.

[15] A. Font Llitjós, "Improving pronunciation accuracy of proper names with language origin classes," Master's thesis, Carnegie Mellon University, Pittsburgh, PA, 2001.

[16] ITU-T Rec. 830, *Subjective Performance Assessment of Telephone-Band and Wideband Digital Codecs*, International Telecommunications Union, Geneva, Switzerland, Feb. 1996.

[17] S. Siegel and N. J. Castellan, *Nonparametric Statistics for the Behavioral Sciences*, 2nd ed. New York, NY: McGraw-Hill, 1988.

[18] Y. Marchand and R. I. Damper, "Can syllabification improve pronunciation by analogy?" 2005, *Natural Language Engineering*, in press.