IMPROVING PRONUNCIATION BY ANALOGY FOR TEXT-TO-SPEECH APPLICATIONS

R.I. Damper *Y. Marchand*

Department of Electronics and Computer Science,
University of Southampton, Southampton SO17 1BJ, UK

ABSTRACT

This paper extends previous work on pronunciation by analogy (PbA) in several directions. PbA is a data-driven method for converting letters to sound, with potential application to next-generation text-to-speech systems. We experiment with a range of methods for matching letter patterns in input words to those in the system dictionary when building a pronunciation lattice. We give preliminary consideration to deriving lexical stress for input words. Common errors are analysed: these mostly involve vowel letters and phonemes. An output is not necessarily guaranteed in PbA – the so-called silence problem. We report on a simple but effective strategy for silence avoidance. Finally, we introduce the idea of using different strategies in combination to improve performance.

1. INTRODUCTION

Modern text-to-speech (TTS) systems use look-up in a large dictionary as the primary strategy to determine the pronunciation of input words. However, it is not possible to list exhaustively all the words of a language, so that a secondary or 'back-up' strategy is required for words not in the system dictionary. Pronunciation by analogy (PbA) is a data-driven technique for the automatic phonemization of text, first proposed over a decade ago by Dedina and Nusbaum [1, 2]. Although initially PbA attracted little attention, several groups around the world are now trying to develop the approach. There is accumulating evidence that PbA easily outperforms linguistic rewrite rules [3, 4, 5, 6] – its main rival and the method traditionally used in past and current TTS systems – and at vastly lower development cost.

In this paper, we extend previous work on PbA in directions which are intended to improve its usability within the pronunciation component of a TTS system. We have studied extended methods both for pattern matching (between the input word and the dictionary entries) and for heuristic path-finding in the pronunciation lattice. This has produced improvements on the best results so far reported in the literature, evaluated on the same lexicon (Webster's dictionary, used by Sejnowski and Rosenberg [7] to train NETtalk), but these improvements have so far been only small. We have also analysed the errors produced, in an attempt to identify common categories of error which might thereby be avoided. Previous implementations of PbA have not always guaranteed an output in all cases – the 'silence problem'. This problem is attacked by a (heuristic) lattice-concatenation method, and this delivers a useful performance improvement.

We also include consideration of lexical stress. There are several ways that stress information might be used. In this work, we restrict consideration to the use of the basic analogy process to map letters to symbols coding stress patterns.

We commence with a brief review of previous work on PbA.

2. PREVIOUS WORKS

2.1. Dedina and Nusbaum's System

Figure 1 shows a block diagram of the Dedina and Nusbaum (D&N) system. The input word is compared to words in the dictionary (*Webster's Pocket Dictionary*). Previously, the letters of each word had been automatically aligned (by a simple Lisp program) with the phonemes specifying the word's pronunciation. Matching substrings are used to build a pronunciation lattice which is then traversed to find the 'best' pronunciation of the input. The 'best' path through the lattice is assessed according to a decision function with two heuristics (shortest path, if unique, or best scoring of the tied shortest paths).

D&N's system was tested on just 70 monosyllabic pseudowords (previously used in reading studies by Glushko [8, 9]). As an illustration, Figure 2 shows the pronunciation lattice for the pseudoword *shead*. Such a test is largely irrelevant to TTS applications: the test set is not representative of general English, either in the small number of words used or their length. Also, the results seem to be difficult (even impossible!) to replicate [3, 4, 6].

2.2. Sullivan and Damper's System

Sullivan and Damper [10] describe a PbA system for English and German. They used a more principled alignment procedure than D&N and a different kind of pronunciation lattice in which nodes correspond to the junctures between letters, rather than to the letters themselves. (The advantage of this is that the system is never 'silent' – unable to produce a pronunciation.) Scoring of the candidate pronunciations used something much closer to a probabilistic

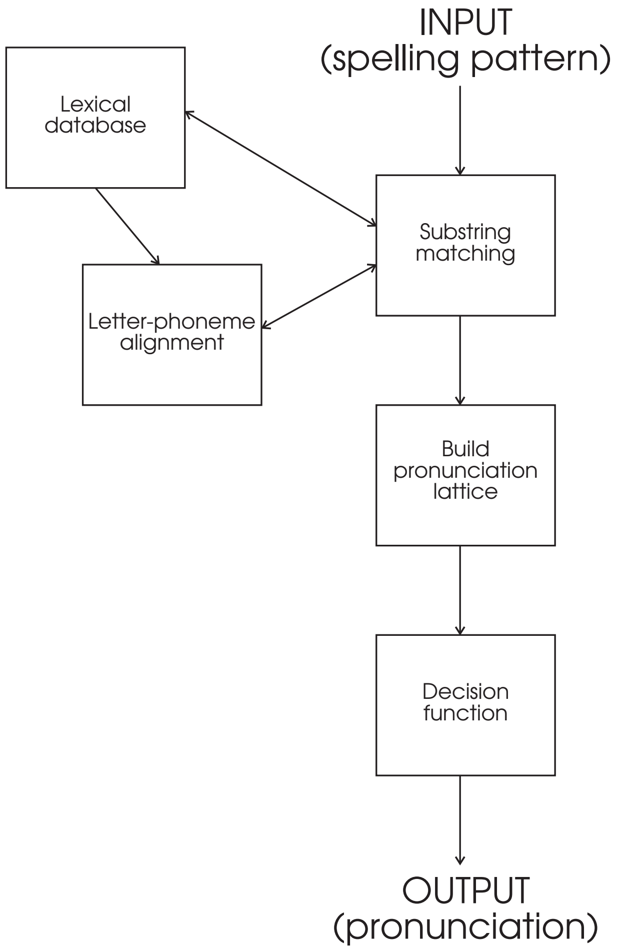

Figure 1: Block diagram of Dedina and Nusbaum's PRONOUNCE.

model than D&N's heuristics. However, testing was again on small sets of pseudowords.

2.3. Damper and Eastmond's System

Damper and Eastmond [3, 4] reimplemented D&N's system but improved the heuristics used to find the best path through the pronunciation lattice. Also, they conducted more realistic and relevant testing on large corpora of real words (approximately 16,000–20,000 words). This was done by removing each word in turn from the lexicon and deriving a pronunciation by analogy with the remaining words. They proposed a method for silence avoidance based on Sullivan and Damper's type of lattice, but much larger lattices resulted and, when using a large test corpus, these could not be searched in reasonable time. No analysis of errors was given. Such analysis is potentially useful in pinpointing and avoiding common sources of error.

2.4. Yvon's System

Recognising that paths through the D&N type of pronunciation lattice imply a (single) shared phoneme between contiguous matching substrings, Yvon [6] generalised this to produce a system (SMPA) based on "unbounded overlapping chunks". Although this was only tested on relatively small subsets (about a tenth) of the lexicon, clear performance improvements were obtained over D&N.

In a later paper, Yvon [11] develops a more formal and linguistic notion of 'analogy'. However, a system based on this notion performs less well than SMPA and is afflicted by a high silence rate.

3. NEW METHODS

In this section, we present our latest developments in PbA. Unless otherwise stated, we have followed the methodology of Damper and Eastmond [4].

3.1. Full Pattern Matching

D&N's pattern matching (when building the pronunciation lattice) is a 'partial' one. That is, the process starts with the left-most letter of the input string and of the current dictionary entry aligned and looks for matching substrings. The shorter of the two strings is then shifted right by one letter and the process repeated. This continues until the two are right-aligned, i.e. the number of right shifts is equal to the difference in length between the two strings. There seems to be no essential reasoning for starting and discontinuing matching at these points. That is, we could shift and match over the range of all possible overlaps.

One conceivable objection to partial pattern matching is that some morphemes could be both prefix and suffix (e.g. someBODY and BODYguard). From this point of view, full matching seems worth consideration. A linguistic justification for the full method is that affixation is often implicated in the creation of new words.

3.2. Other Pattern Matchings

The pattern matching approaches considered so far use little information other than the individual letter identities. It seems at least worth considering the use of other knowledge, such as whether letters are vowels or consonants, and surrounding letters (context). We have considered the following methods:

Length: this simply means that only words of the same length are matched.

Vowel & Consonant: this means that any matched substring must contain at least one vowel and one consonant. The (naïve) hypothesis is that the consonant(s) associated with a vowel are responsible for the vowel sound.

Surrounding consonant(s): pattern matching for word-medial vowels has to include the surrounding consonants. The idea comes from the error analysis (see section 4.2) which shows that the vowels are mainly responsible for the errors.

Stress: see below.

Syllable Boundaries: see below.

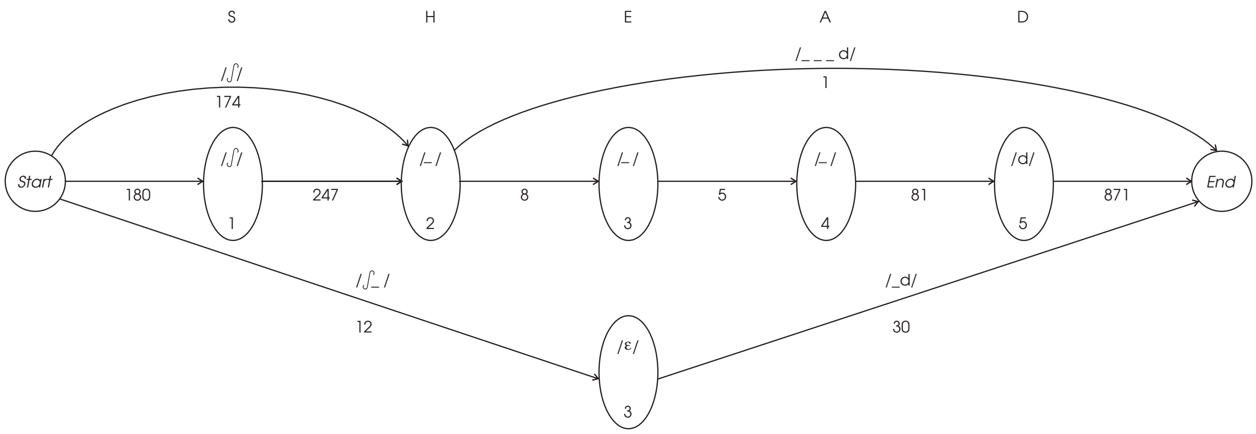

Figure 2: Pronunciation lattice for the pseudoword *shead*.

The latter two types of information are obtained from the NETtalk dictionary. This has data arranged in columns:

aardvark	`a-rdvark`	1 <<<> 2 <<
aback	`xb@k-`	0 > 1 <<
abacus	`@bxkxs`	1 < 0 > 0 <
abaft	`xb@ft`	0 > 1 <<

Here the second column is the pronunciation, manually aligned by Sejnowski and Rosenberg [7]. (The '–' symbol is the null phoneme, introduced to give a strict one-to-one mapping between letters and phonemes.) The third column encodes the syllable boundaries for the words and their corresponding stress patterns, where (according to these authors):

<	denotes	Syllable Boundary
>	"	Syllable Boundary
1	"	Primary Stress
2	"	Secondary Stress
0	"	Tertiary Stress

From this, we infer four rules (or regular expressions) to identify syllable boundaries:

R1: [<>]
R2: [< digit]
R3: [digit >]
R4: [digit digit]

Table 1 gives some examples of syllable boundaries and decoded stress pattern obtained using these rules.

For the pattern matching using syllable information, two methods have been used:

1. Pattern matching is forbidden across syllable boundaries. We call this *Syl.1*.

2. Whole syllables are used as the 'atomic' units for pattern matching in place of letters. We call this *Syl.2*.

It is important to note that with the *Syl.2* method, we are using syllabification information about the input word that in the real case, we do not have. The justification for this is to determine if this information is helpful. If it is, it would make an attempt at automatic syllabification of the input worthwhile.

There are several ways for the stress information to be used. In one approach, we can find a stress pattern by analogy – in precisely the same way as we find pronunciations. The only difference is that in one case we are mapping from letters to symbols from the set of phonemes and in the other we map to symbols from the set $\{0,1,2\}$. We could also attempt to use stress patterns in the lexicon as part of the process of finding a pronunciation. Alternatively, we might turn this around and attempt to use the obtained pronunciation as part of the process of finding the stress pattern. In this paper, we present results only for the first of these approaches – finding the stress independent of the pronunciation. (Work on the other possibilities is ongoing.)

The segments take the 'digit stress'. By this, we mean that the same stress-level digit code (0, 1 or 2) is given to all letters within a syllable. There are no '<' or '>' codes in the pattern.

3.3. Path finding

The last research direction has been to give further consideration to D&N's second heuristic – namely, the way of resolving ties between candidate pronunciations with equal shortest path lengths. Thus, all our strategies apply to the shortest path. They are:

Sum: sum of the arc frequencies.

Product: product of the arc frequencies.

Total Product: sum of the product of the arc frequencies for all shortest paths giving the same pronunciation. (Note that this is slightly different from Damper and Eastmond's 'TP', which considered *all* paths giving the same pronunciation, whether they were shortest paths or not.)

Table 1: Examples of stress and syllabification patterns.

<table>
<tr><th>Word</th><th>abbreviate</th><th>abecedarian</th><th>actuarial</th></tr>
<tr><td>Stress pattern</td><td>0 <>> 1 > 02 <<</td><td>2 > 0 > 0 > 1 < 00 <</td><td>2 <> 01 < 00 <</td></tr>
<tr><td>Syllables</td><td>ab|bre|vi|ate</td><td>a|be|ce|dar|i|an</td><td>ac|tu|ar|i|al</td></tr>
<tr><td>Stress</td><td>00|111|00|222</td><td>2|00|00|111|0|00</td><td>22|00|11|0|00</td></tr>
</table>

Weighted Total Product: this is like the Total Product score, except that each contribution to the total product is weighted by the number of letters associated with each arc.

4. RESULTS

4.1. Lexical Words

Homonyms (413 entries) and one-letter words (*a* and *o*) (which are obviously problematic for our PbA system) were removed from the original NETtalk corpus. This left 19,594 words.

Table 2 shows the results obtained with the various methods of pattern matching and heuristic scoring. These show that best results are found with the full pattern matching. These figures are only marginally (but consistently) better that those obtained by Damper and Eastmond[4, p. 17)] who, following D&N, used partial pattern matching. Our best result (61.7% words correct) is for the weighted total product score, but the improvement over the total product score is slight. We certainly could not claim statistical significance for this difference.

The other methods of pattern matching did less well. None of them really approached the full matching method. The Vowel&Consonant method was the best of the remainder (58.0% words correct). Indeed, results for Syl.2 are very low, at 5.3% words correct. Such a low figure is only to be expected, since the severe constraints placed on matching naturally produce a high proportion of 'silence'. This raises the question of why such a limited technique was used in the first place.

The answer is that we anticipated using the methods 'in cascade'. That is, if one technique fails to produce a pronunciation (i.e. gives silence), we try another. Thus, a particular technique may work poorly in general, yet still be worth using in cascade. To illustrate this, Syl.2 works badly in general but in just those cases where it is applicable (no silence), it achieves 73.0% words correct. This is actually higher than any of the other strategies.

Table 3 shows, for completeness, some statistics related to pronunciation lattice size (and silence production) for the various methods.

We now consider the results of finding the digit stress pattern by analogy. At this stage, these were obtained using full pattern matching and the Weighted Total Product scoring. (This is true of all the following results in this paper.) This was chosen as the best-performing combination in Table 2. (However, we recognise that the other approaches may work as well or even better.) In only 54.6% of cases were totally correct stress patterns obtained (corresponding to 75% digits correct). This is slightly surprising, as the set of digit symbols is very small (just 3) implying a rather simple mapping problem. This warrants further investigation. We also determined the percentage of words (41.8%) in which both pronunciation and stress are correct.

4.2. Analysis of errors

Table 4 identifies the main sources of pronunciation error. The 'Words' entry indicates, in rank order, the 10 letters in the input which were most often mapped to an incorrect phoneme. In the orthographic domain, the commonest problem is mispronunciation of letter *A* which produces 20.9% of errors. To some extent, this is a natural consequence of the high frequency of this letter in English. The 'Prop.' entry shows that 9.0% of the letters in words which the PbA system could pronounce were *A*'s. The 'Output' entry details the 10 commonest incorrect phonemes in the system's output. The schwa vowel accounts for 21.1% of errors in this case. Again, this reflects the extremely common occurrence of this phoneme. Finally, the 'Desired' entry shows the 10 phonemes in the correct pronunciation which most often received a wrong translation. These are the same as for the 'Output' entry, but in slightly different order. It is clear that, overall, vowel errors vastly outnumber consonant errors. The null phoneme is also problematic.

Table 5 shows the phoneme substitution errors seen. Again, schwa is the main cause of problems.

4.3. Silence Avoidance

We have experimented with a very simple strategy for silence avoidance. If there is just a single break in the lattice, the pronunciations for the two halves are concatenated. This was suggested by an examination of the silence errors.

65 words gave silence. Among these, approximately 45% were clearly composite words (e.g. *surfboard*) and 43% were words of foreign origin. After concatenation, 43.1% of the silence words received a correct pronunciation. Table 6 shows some examples. The usual cause of a break in the pronunciation lattice is a rare digram – occurring only in the word itself. Once it is removed from the lexicon, it becomes impossible to produce a connected lattice. 6 of the words in Table 6 are in this category.

When the full pattern matching with weighted total

Table 2: Results (% correct) for PbA with different pattern matching methods and heuristics. TP denotes Total Product; WTP denotes Weighted Total Product.

Methods	Sum		Prod.		TP		WTP	
	Words	Phon.	Words	Phon.	Words	Phon.	Words	Phon.
Full	58.4	90.7	59.1	91.0	61.5	91.5	61.7	91.6
Length	41.8	79.8	43.0	80.1	43.9	80.3	43.8	80.3
Surr. Conson.	52.2	85.1	52.2	85.1	53.1	85.4	53.8	85.5
Syl.1	32.5	77.7	32.8	78.0	33.2	78.1	33.0	78.0
Syl.2	5.2	8.3	5.3	8.3	5.3	8.3	5.3	8.3
Vow.& Cons.	54.4	87.3	56.1	87.7	58.0	88.0	58.0	88.1

Table 3: Features of the pronunciation lattices.

Features	Methods					
	Full	Length	Surr.C	Syl.1	Syl.2	V&C
Num. of nodes	44	35	15	42	11	41
Num. of arcs	169	86	38	112	15	150
Num. of P. Matchings	10603	1320	2981	8958	678	6381
Silence (%)	0.3	7.4	6.6	5.6	92.7	5.3

Table 4: Main factors responsible for errors.

Words	Letters	A	E	O	I	U	R	L	N	S	T	Total
	Prop. (%)	9.0	11.0	6.9	8.8	3.8	7.5	5.5	6.8	5.3	7.7	72.3
	Errors (%)	20.9	20.0	16.3	14.4	6.9	4.2	3.3	2.7	2.7	1.9	93.2
Output	Phonemes	x	I	–	@	E	a	i	o	e	A	Total
	Prop.(%)	8.0	5.0	14.7	2.8	2.5	1.9	3.0	1.5	1.8	1.1	42.3
	Errors(%)	21.1	9.4	8.2	7.4	7.2	5.9	4.3	3.5	3.3	2.9	73.2
Desired	Phonemes	x	–	I	a	E	i	@	e	A	o	Total
	Prop. (%)	8.1	14.8	4.8	1.9	2.4	3.1	2.7	1.8	1.2	1.5	42.3
	Errors (%)	22.4	9.3	7.5	6.2	5.6	5.4	5.3	4.2	4.1	3.7	73.8

Table 5: Ranked substitution errors.

Desired	x	@	a	x	x	x	I	E	x	R	Total
Output	@	x	x	I	a	E	x	x	–	r	
Percentage	4.3	3.4	3.3	3.2	2.9	2.5	2.5	2.4	2.3	2.2	29.1

Table 6: Example results of the concatenation strategy.

Word	Digram	Pronunciation	Concatenation
thoroughgoing	1 × hg	`T--Ro---goIG-`	OK
surfboard	2 × fb	`s-Rfbor-d`	OK
earthquake	1 × hq	`--RTkw-ek-`	OK
blitzkrieg	1 × zk	`blItskri-g`	`blItskrixg`
halfpenny	1 × fp	`he--pxn-i`	`h@-fpEn-i`
flapjack	1 × pj	`fl@pJ@k-`	OK
misjudge	1 × sj	`mIsJ^-J-`	OK

product strategy is combined with lattice-repair by concatenation in cascade, we obtain 61.9% words correct and 91.8% phonemes correct. This is (marginally) the best score we have obtained.

5. CONCLUSIONS

We have extended previous work in pronunciation by analogy (PbA) in several directions. Principally, we have experimented with a range of pattern matching strategies. None works better than full pattern matching. In the near future, however, we intend to use multiple strategies for producing pronunciations. We will then use evidence combination techniques to derive an overall pronunciation. The hope is that, by using the pooled evidence, we can produce better pronunciations than any single technique.

A weighted 'total product' scoring heuristic has been used: improvements were only marginal. We should work towards a proper probabilistic model.

We have also attempted to produce stress patterns for input words, but results are provisional at this stage. Much remains to be done on this.

Finally, we have analysed common errors of pronunciation. Future work should attempt to find an effective procedure for dealing with the vowels, which are often responsible for bad pronunciations.

ACKNOWLEDGEMENT

This work was funded by the UK Economic and Social Research Council via research grant R000235487: "Speech Synthesis by Analogy".

6. REFERENCES

1. M. J. Dedina and H. C. Nusbaum. PRONOUNCE: a program for pronunciation by analogy, 1986. Speech Research Laboratory Progress Report No. 12.
2. M. J. Dedina and H. C. Nusbaum. PRONOUNCE: a program for pronunciation by analogy. *Computer Speech and Language*, 5:55–64, 1991.
3. R. I. Damper and J. F. G. Eastmond. Pronouncing text by analogy. In *Proceedings of 16th International Conference on Computational Linguistics*, volume 2, pages 268–273, Copenhagen, Denmark, 1996.
4. R. I. Damper and J. F. G. Eastmond. Pronunciation by analogy: impact of implementational choices on performance. *Language and Speech*, 40(1):1–23, 1997.
5. R. I. Damper and K. Gustafson. Evaluating the pronunciation component of a text-to-speech system. In *Speech and Language Technology (SALT) Club Workshop on Evaluation in Speech and Language Technology*, pages 72–79, Sheffield, UK, 1997.
6. F. Yvon. Grapheme-to-phoneme conversion using multiple unbounded overlapping chunks. In *Proceedings of Conference on New Methods in Natural Language Processing (NeMLaP '96)*, pages 218–228, Ankara, Turkey, 1996.
7. T. J. Sejnowski and C. R. Rosenberg. Parallel networks that learn to pronounce English text. *Complex Systems*, 1:145–168, 1987.
8. R. J. Glushko. The organization and activation of orthographic knowledge in reading aloud. *Journal of Experimental Psychology: Human Perception and Performance*, 5:674–691, 1979.
9. R. J. Glushko. Principles for pronouncing print: The psychology of phonography. In A. M. Lesgold and C. A. Perfetti, editors, *Interactive Processes in Reading*, pages 61–84. Lawrence Erlbaum, Hillsdale, NJ, 1981.
10. K. P. H. Sullivan and R. I. Damper. Novel-word pronunciation: A cross-language study. *Speech Communication*, 13:441–452, 1993.
11. F. Yvon. Paradigmatic cascades: A linguistically sound model of pronunciation by analogy. In *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics*, pages 429–435, Madrid, Spain, 1997.