Evaluating Automatic Syllabification Algorithms for English

Yannick Marchand[1,2]*, Connie R. Adsett*[1,2] *and Robert I. Damper*[1,3]

[1]Institute for Biodiagnostics (Atlantic), National Research Council Canada,
1796 Summer Street, Suite 3900,
Halifax, Nova Scotia, Canada B3H 3A7

[2]Faculty of Computer Science, Dalhousie University,
Halifax, Nova Scotia, Canada B3H 1W5

[3]School of Electronics and Computer Science
University of Southampton, Southampton SO17 1BJ, UK

{yannick.marchand, connie.adsett}@nrc-cnrc.gc.ca, rid@ecs.soton.ac.uk

Abstract

Automatic syllabification of words is challenging, not least because the syllable is difficult to define precisely. This task is important for word modelling in the composition process of concatenative synthesis as well as in automatic speech recognition. There are two broad approaches to perform automatic syllabification: rule-based and data-driven. The rule-based method effectively embodies some theoretical position regarding the syllable, whereas the data-driven paradigm infers 'new' syllabifications from examples assumed to be correctly-syllabified already. This paper compares the performance of the two basic approaches. However, it is difficult to determine a correct syllabification in all cases and so to establish the quality of the 'gold standard' corpus used either to quantitatively evaluate the output of an automatic algorithm or as the example-set on which data-driven methods crucially depend. Thus, three lexical databases of pre-syllabified words were used. Two of these lexicons hold the same 18,016 words with their corresponding syllabifications coming from independent sources, whereas the third corresponds to the 13,594 words that share the same syllabifications according to these two sources. As well as one rule-based approach (Fisher's implementation of Kahn's syllabification theory), three data-driven techniques are evaluated: a look-up procedure, an exemplar-based generalization technique, and syllabification by analogy (SbA). The results on the three databases show consistent and robust patterns: the data-driven techniques outperform the rule-based system in word and juncture accuracies by a very significant margin and best results are obtained with SbA.

1. Introduction

The syllable has been much discussed as a linguistic unit. Whereas some linguists make it central to their theories (e.g., [1, 2]), others have ignored it or even argued against it as a useful theoretical construct (e.g., [3]). Much of the controversy centers around the difficulty of defining the syllable. Crystal [4], for instance, states that the syllable is "[a] unit of pronunciation typically larger than a single sound and smaller than a word" but goes on to write: "Providing a precise definition of the syllable is not an easy task" [p. 342]. There is general agreement that a syllable consists of a *nucleus* that is almost always a vowel, together with zero or more preceding consonants (the *onset*) and zero or more following consonants (the *coda*). However, determining exactly which consonants of a multisyllabic word belong to which syllable is problematic. Good general accounts of the controversy are provided by [5] and [6], with the former more specifically considering English—the language of interest in this paper—and the latter focusing on French.

However it is defined, and whatever the rights or wrongs of theorising about its linguistic status, syllable knowledge aids word modeling in automatic speech recognition and/or the unit selection and composition process of concatenative synthesis. For instance, Müller, Möbius and Prescher [7] write "syllable structure represents valuable information for pronunciation systems" [p. 225]. That is, the pronunciation of a phoneme can depend upon where it is in a syllable and therefore there are good practical reasons for seeking powerful algorithms to syllabify words.

Traditional approaches to automatic syllabification have been *rule-based* (or knowledge-based), implementing notions such as the maximal onset principle [1, 8] and sonority hierarchy [9], including ideas about what constitute phonotactically legal sequences in the coda, for instance. An alternative to the rule-based methodology is the *data-driven* (or corpus-based) approach, which attempts to infer 'new' syllabifications from an evidence base of already-syllabified words (i.e., a dictionary or lexicon[1]).

This paper compares the performance of these two basic approaches to automatic syllabification in the pronunciation domain. Our work attempts to be *predictive*, aimed at finding good syllabifications for practical application in speech technology and computational linguistics, rather than *descriptive*, aimed at explaining experimental data and/or giving insight into any linguistic theory of the syllable.

2. Electronic lexical databases

A key issue in assessing algorithms for automatic syllabification is the quality of the 'gold standard' corpus used to define the correct result. Further, in the data-driven paradigm, this corpus forms the evidence base for inferring new syllabifications;

[1]In this paper, we will use the terms *evidence base*, *lexical database*, *dictionary*, *corpus*, and *lexicon* interchangeably, except where we refer to a 'dictionary' by name (e.g., *Webster's Pocket Dictionary*).

hence, it is vital that its content is accurate. This, however, is extremely difficult due to the absence of any means of determining canonically correct syllabifications. Our approach is to use multiple dictionaries and to seek consensus among them, so as to reduce the possibility that our results are affected by the choice of a particular, idiosyncratic corpus.

In this work, we used two public-domain dictionaries—*Webster's Pocket Dictionary* and the *Wordsmyth English Dictionary-Thesaurus*—as the sources from which we derive three lexical databases, as described below.

2.1. Webster's Pocket Dictionary

The primary lexical database in this work is *Webster's Pocket Dictionary* (20,009 words), as used by [10] to train their NETtalk neural network. The database is publicly available for non-commercial use from `ftp://svr-ftp.eng.cam.ac.uk/pub/comp.speech/dictionaries/` (last accessed 11 May 2007). For consistency with our previous work on pronunciation using this dictionary, homonyms (413 entries) were removed from the original NETtalk dataset leaving 19,596 entries. Sejnowski and Rosenberg have manually aligned the data, to impose a strict one-to-one correspondence between letters and phonemes[2]. The phoneme inventory is of size 51, including the null phoneme and 'new' phonemes (e.g., /K/ and /X/) invented to avoid the use of null letters when one letter corresponds to two phonemes, as in <x> → /ks/. The null phoneme (represented by the '–' symbol) was introduced to give a strict one-to-one alignment between letters and phonemes to satisfy the training requirements of NETtalk. In this paper, we retain the use of the original phonetic symbols (see [10], Appendix A, pp. 161–162) rather than transliterating to the symbols recommended by the International Phonetic Association. We do so to maintain consistency with this publicly-available lexicon.

In addition to the pronunciation, Sejnowski and Rosenberg have also indicated stress and syllabification patterns for each word. The form of the data is:

accumulate	`xk-YmYlet-`	0<>1>0>2<<
adaptation	`@d@pteS-xn`	2<2<>1>0<<

The second column is the pronunciation and the third column encodes the syllable boundaries for the words and their corresponding stress patterns:

<	denotes	syllable boundary (right)
>	"	syllable boundary (left)
1	"	primary stress
2	"	secondary stress
0	"	tertiary stress

Stress is associated with vowel letters and arrows with consonants. The arrows point towards the stress nuclei and change direction at syllable boundaries. To this extent, "syllable boundary (right/left)" is a misnomer because this information is not adequate by itself to place syllable boundaries directly. We can, however, infer four rules (or regular expressions) to identify syllable boundaries. Denoting boundaries by ' | ':

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

[2]See [11] for extensive discussion of this alignment process and an algorithm for doing it automatically.

Word	*accumulate*	*adaptation*
Stress pattern	0<>1>0>2<<	2<2<>1>0<<
Syllabification	ac | cu | mu | late	ad | ap | ta | tion
Digit stress	00 | 11 | 00 | 2222	22 | 22 | 11 | 0000

Table 1: *Examples of stress and syllabification patterns.*

These have been confirmed as correct by Sejnowski (personal communication). Table 1 gives the syllable patterns of the three above examples.

2.2. Wordsmyth English Dictionary-Thesaurus

Disagreements may exist about the way a word should be segmented into syllables. A second (independent) lexical source was therefore used, namely the *Wordsmyth English Dictionary-Thesaurus*, so that our results would not be overly specialized to one particular dictionary. This source is also available via the World Wide Web from `www.wordsmyth.net` (last accessed 11 May 2007). This on-line lexical database originated in the early 1980's when Robert Parks, a Fulbright Fellowship researcher in Japan, began to develop an English dictionary for students to use on their computers. In 1991 and 1992, the dictionary was licensed to IBM to integrate into their products, and IBM in turn supported the development of the associated thesaurus. In 1996, the University of Chicago's ARTFL (American and French Research on the Treasury of the French Language) Project assisted in presenting the first World Wide Web edition. The dictionary is composed of about 50,000 headwords covering all areas of knowledge without technical vocabulary. It provides the syllables, pronunciation, part of speech, inflected forms, and definition for each word.

2.3. The three lexical databases

Homonyms were removed from the original *Webster's Pocket Dictionary* leaving 19,596 entries. Of these words, 18,016 were also found in the *Wordsmyth English Dictionary-Thesaurus*. These two independant dictionaries, each consisting of 18,016 syllabified entries, are referred to as *S&R* and *Wordsmyth*, respectively. A third database of syllabified words (hereafter *Overlap*) was derived consisting of the 13,594 words present in both public-domain dictionaries with identical syllabification patterns in these two independent lexical sources.

3. Syllabification algorithms

In this section, we briefly describe the four automatic syllabification techniques for which performance was compared.

3.1. Fisher's implementation of Kahn's procedure

In his PhD dissertation, Kahn proposed a theory of syllabification based on a different type of constraint [8]. Kahn postulated that syllabification in English is derived from three categories of consonant clusters: possible syllable-initial, possible syllable-final and 'universally-bad' syllable-initial (in his terminology). These consonant clusters are derived from the beginnings and endings of existing English words. For example, the two-phoneme sound /br/ is a possible syllable-initial consonant cluster because it forms the beginning of the word pronunciation /bred/ (<bread>) and it is therefore possible to syllabify the pronunciation /ənbreɪd/ (<unbraid>) as

/ən | breɪd/. By contrast, /rk/ is considered a universally-bad syllable-initial consonant cluster because no English word begins with this sound combination. Therefore the pronunciation /markət/ (<market>) would be syllabified as /mar | kət/ and not /ma | rkət/.

A C implementation of Kahn's theory was developed in 1996 by William Fisher and can be downloaded from the file: `ftp://jaguar.ncsl.nist.gov/pub/tsylb2-1.1.tar.Z` (last accessed 11 May 2007). Because we were interested in the standard syllabification, we selected the two most appropriate of the five speech rates available in the program, the "slow, over-precise" (hereafter Basic) and the "ordinary conversational speech" (hereafter OCS) rates. The program also allowed the unsyllabified input to be provided with stress information (primary, secondary and no stress) on some specific phonemes[3] and without stress information. We processed the word list both ways, using the stress information provided in *S&R* (i.e., the digit stress—see Table 2.1). The phoneme set used in his program was translated to the phoneme set of *S&R* and all instances of the null phoneme were also removed because this special 'phoneme' was not part of Fisher's set.

3.2. Syllabification by analogy

Syllabification by analogy closely follows the principles of pronunciation by analogy (PbA) set out in detail in our earlier publications [12, 13, 14, 15]. In PbA, when an unknown word is presented as input to the system, so-called full pattern matching 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 below. One of the two strings is then shifted relative to the other by one letter and the matching process continues, 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 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, five different scoring strategies are applied and the winning candidate selected on the basis of their rank [13, 14].

The major modification in converting PbA to SbA is to represent all junctures between phonemes explicitly. This representation must be different in the case of:

1. input words, where the syllabification is unknown;
2. lexical entries, where it is known;
3. the SbA output, where it is inferred.

For example, the input pronunciation /@bi/[4] (<abbey>) is expanded to /@ ∗ b ∗ i/. Here the '∗' symbol merely indicates the *possibility* of a syllable boundary. On the other hand, a dictionary entry such as /@bncrmL/ <abnormal> is expanded to /@ ∗ b | n ∗ c ∗ r | m ∗ L/. In this case, the '∗' symbols indicate the known absence of a syllable boundary. During pattern matching, '∗' in the input is allowed to match either with '∗' or with '|' in the dictionary entries. A '∗ − ∗' match is entered into the syllabification lattice as a '∗' whereas a '∗ − |' match is entered into the syllabification lattice as a '|'. The syllabification lattice has exactly the same form as the pronunciation lattice, except that '∗' is explicitly represented as an input symbol (labelling nodes), '∗' and '|' are explicitly represented as possible output symbols (labelling arcs), and there is no pronunciation information labelling the nodes and arcs. From here, the process proceeds exactly as for PbA, eventually producing as output a syllabified version of /@bi/ <abbey> such as /@ ∗ b | i/, from which the '∗' symbols are removed to yield the final output /@b | i/. The modifications to perform SbA in the pronunciation domain should now be obvious.

In our previous syllabification work using analogy [15], we obtained best results by combining only 3 of the 5 scoring strategies when choosing between tied shortest paths. These were the product of arc frequencies, the frequency of the same pronunciation, and the 'weak link' (see [13] and [14] for full specification). Accordingly, in this work, these same three scoring strategies are used exclusively, and combined by rank fusion, for SbA.

3.3. Look-up procedure

This method was originally proposed by [16] as a means of letter-to-phoneme conversion (i.e., automatic pronunciation), where it was shown to be superior to NETtalk, the well-known neural network [10]. It was then adapted for the syllabification process and presented in the comparison of syllabification algorithms for Dutch spellings by [17]. The first step is to construct a table encoding the knowledge implicit in the training set by converting each syllabified entry into a series of N-grams. Each N-gram has a left and right context and a central, 'focus' character. The length of the N-gram (i.e., N) is equal to the sum of the sizes of the left and right contexts plus one (the focus character).

For example, if the syllabified word /KId | ni/ (<kid | ney>) is part of the training corpus, then with a left context of 1 character and a right context of 2 characters, the N-grams (or 4-grams in this case) for this word would be: <− KId>, <KIdn>, <Idni>, <dni −>, and <ni − −>. That is, to allow every character to be a focus character, there is an N-gram for each character in a word. When the focus character has no left context

[3] . . . designated as syllabic by Fisher. These are: 'ux', 'ih', 'ix', 'ey', 'eh', 'ae', 'aa', 'aax', 's', 'ao', 'ow', 'uh', 'uw', 'ay', 'oy', 'aw', 'er', 'axr', 'ax', 'ah', 'el', 'em', and 'en' using his phoneme notation.

[4] Examples from this point on use the phoneme set from *Webster's Pocket Dictionary*.

(as in <−KId>) or right context (as in <ni− −>), the character positions in the context are filled with null characters. Each N-gram is stored in the table along with the corresponding juncture class, i.e., the syllabification information.

Once the construction of the look-up table is complete, words for which the syllabification is unknown can be syllabified based on the information in the table. Input words are broken down into a set of N-grams in the same manner described above for table construction. The table is then searched for the closest matches to each N-gram. When found, closest matches are examined to determine whether the majority has, or does not have, a syllable boundary following the focus character. If the majority has a syllable boundary, a syllable boundary is placed at the appropriate position in the word; otherwise, a non-syllable boundary is placed at that position.

The process of determining which N-grams in the pre-compiled look-up table fit best a given N-gram is described in Algorithm 1. Here, `NgramT` is a given N-gram stored in the table and `NgramS` is an N-gram to be syllabified. It follows that `NgramT[i]` is the ith position in the N-gram (for example, `NgramT[1] = m` when `NgramT` is <midn>). The closest-fit N-grams are those with the highest `MatchValue`.

Algorithm 1 : Computation of best-fit N-gram in the look-up procedure.

```
FindMatchValue(weights, NgramT, NgramS)
MatchValue := 0
for i := 1 to length(weights) do
  if (NgramT[i] = NgramS[i]) then
    MatchValue := MatchValue +
    weights[i]
  end if
end for
```

We ran the look-up procedure using all 15 different sets of weights presented in the original description of the method [16].

3.4. Exemplar-Based Generalization

The version tested here (also known as IB1-IG) is due to [18]. It operates in a manner similar to the look-up procedure with the only difference being the weights used to determine the closest-fit N-grams. In this method, the weights are calculated with a function that determines the relative importance of each position in the N-gram (i.e., phoneme positions). The process of determining the weights is based on the concept of information entropy by using information from the table of stored N-grams. Each position in an N-gram is considered to contribute a real-valued amount of information to the process of determining the placement of a syllable boundary. This value can be determined via the series of steps presented below.

First, the entropy of the entire table of N-grams extracted from the training corpus is calculated. Essentially, Daelemans, van den Bosch and Weijters define database (or look-up table) information entropy as "the number of bits of information needed to know the decision [whether a syllable boundary should be placed after the focus character or not] of a database given a pattern [or N-gram]." This is calculated as:

$$E(D) = -\sum_{i=1}^{2} P_i \log_2 P_i \tag{1}$$

where $E(D)$ is the information entropy of database D, P_1 is the probability of an N-gram being associated with a syllable-boundary decision, and P_2 is the probability of an N-gram being associated with a non-syllable-boundary decision. As there are only two possibilities—to place or not to place a syllable boundary after the focus character—equation (1) can also be written as:

$$E(D) = -\alpha \log_2 \alpha + \beta \log_2 \beta \quad \text{where } \alpha = \frac{N_S}{N_T} \text{ and } \beta = \frac{N_{\neg S}}{N_T} \tag{2}$$

where N_S is the number of stored N-grams that have a syllable boundary following the focus character, $N_{\neg S}$ is the number of stored N-grams that do not have a syllable boundary following the focus character, and N_T is the number of stored N-grams (i.e., $N_S + N_{\neg S}$).

From equation (2), the information gain of each position in an N-gram can now be determined. This requires two additional equations. The first computes the average information entropy at position f in an N-gram, $E(D_f)$, by taking the "information entropy of the database [or table] restricted to each possible value [or character] for the [position in the N-gram]." This is given by:

$$E(D_f) = \sum_{c \in V} E(D_{f=c}) \frac{\text{card}(D_{f=c})}{\text{card}(D)}$$

where $D_{f=c}$ is the set of those N-grams in the table that have character c at position f, V is the set of characters that occur at position f in a N-gram, and card( ) is the cardinality of a set (i.e., card(D) is the total number of N-grams in database D).

The second equation necessary for calculating the information gain $G(f)$ at a given position f in an N-gram is:

$$G(f) = E(D) - E(D_f)$$

To run this method, we first followed Daelemans, van den Bosch and Weijters and used the same values of N as in their work, namely 3, 5 and 7 with the focus letter in the middle of the N-gram. In addition to these values, we extended the study to use N-grams of size 9 and 11 (with left and right contexts of 4 and 5 respectively).

4. Results

For the rule-based method, there is no difficulty in evaluating syllabification performance on each of the three datasets in their entirety. For data-driven methods, we use the well-established leave-one-out procedure, whereby each word is removed from the corpus in turn, and its syllabification inferred from the remaining words.

Tables 2, 3 and 4 show the results for the various automatic syllabification methods on the *S&R*, *Wordsmyth* and *Overlap* databases. For table look-up, the three sets of weights which provided the best results (for each dictionary) are presented in the tables[5]. Results were obtained for N-grams from $N = 5$ up to $N = 11$ for the exemplar-based approach. As expected, results were poor for $N = 3$ as insufficient context is captured around the focus phoneme, and by $N = 11$ the algorithm indicates that performance was falling off. For the Fisher/Kahn system, there was no difference between the results when stress was provided and when it was not for the Basic (slow) rate of

[5]Version 8:[1,4,16,4,2]; Version 10:[1,4,16,64,16,5,1]; Version 11:[1,4,16,64,256,64,17,4] and Version 13:[4,16,64,256,64,17,4,1].

Algorithm	Accuracy Word	Juncture	|	*
Fisher/Kahn				
Basic	54.23	78.93	62.63	85.34
OCS	54.14	77.47	59.84	84.41
OCS with stress	68.97	86.41	75.65	90.64
SbA	88.53	96.02	92.29	97.50
Look-up Table				
1st, version 10	80.20	94.95	90.51	96.70
2nd, version 8	79.75	94.90	90.49	96.63
3rd, version 13	79.40	94.78	89.90	96.70
Exemplar-based				
$N = 5$	76.47	94.17	87.54	96.79
$N = 7$	79.37	94.80	88.92	97.11
$N = 9$	79.36	94.78	89.04	97.04
$N = 11$	79.10	94.71	88.91	96.99

Table 2: *Syllabification results (percentage correct) on the* S&R *database for word and juncture accuracy.*

Algorithm	Accuracy Word	Juncture	|	*
Fisher/Kahn				
Basic	58.02	81.34	67.04	86.91
OCS	52.58	75.93	57.16	83.24
OCS with stress	63.37	83.40	70.49	88.43
SbA	85.88	94.87	90.32	96.64
Look-up Table				
1st, version 10	75.71	93.41	87.93	95.54
2nd, version 8	75.37	93.36	87.96	95.47
3rd, version 11	74.86	93.26	87.44	95.53
Exemplar-based				
$N = 5$	72.92	92.81	85.04	95.84
$N = 7$	74.92	93.17	85.76	96.05
$N = 9$	82.90	95.54	89.23	97.71
$N = 11$	74.87	93.12	85.86	95.96

Table 3: *Syllabification results (percentage correct) on the* Wordsmyth *database for word and juncture accuracy.*

speech. However, this was not the case for the ordinary conversational speech condition, where the inclusion of stress improves the results.

Results are remarkably consistent across dictionaries. The rule-based method (Fisher/Kahn) is much worse than the data-driven methods. We do not think such a rule-based method is valuable in computational linguistics and/or speech technology. In regards to the data-driven methods, it is difficult to choose between the best table look-up and exemplar-based results although the former does better on two of the three dictionaries. The most striking result, however, is the obvious superiority of SbA.

These tables also show junctures-correct performance overall, as well as the percentages of correct syllable (|) and non-syllable (*) identifications. For all methods, non-syllable boundary identification is less error prone than syllable boundary detection. It seems that all methods are conservative in their placement of syllable boundaries, which are rarer than non-syllable boundaries, resulting in a preponderance of false negative errors over false positives.

Algorithm	Accuracy Word	Juncture	|	*
Fisher/Kahn				
Basic	63.40	83.54	67.80	88.97
OCS	60.90	79.68	60.07	86.44
OCS with stress	74.42	88.14	76.56	92.13
SbA	91.08	96.82	92.90	98.17
Look-up Table				
1st, version 10	83.66	95.74	90.58	97.52
2nd, version 8	83.60	95.76	90.66	97.52
3rd, version 11	82.71	95.55	89.99	97.47
Exemplar-based				
$N = 5$	81.26	95.20	88.51	97.50
$N = 7$	83.12	95.56	89.28	97.73
$N = 9$	82.90	95.54	89.23	97.71
$N = 11$	82.87	95.52	89.23	97.69

Table 4: *Syllabification results (percentage correct) on the* Overlap *database for word and juncture accuracy.*

5. Conclusions

Automatic syllabification is an important but difficult problem that has implications on pronunciation generation for text-to-speech synthesis and pronunciation modeling in speech recognition. There are essentially two possible approaches to automatic syllabification: rule-based and data-driven.

In this work, we have compared one rule set based on expert knowledge and three data-driven methods based on automatic inference from a corpus of already-syllabified words. In the latter case, the issue of a gold standard arises. We attempt to address this by using two independent dictionaries of syllabified words. We also use the 'overlap' or conjunction of entries in the different dictionaries as a separate corpus which ought to be closer to a gold standard than either of the individual contributors, since it does not include words on which they disagree. In this work, we have used two independent dictionaries (*S&R* and *Wordsmyth*) and their overlap. The four methods studied are the rule set from Fisher/Kahn, a table look-up method developed by Weijters, the exemplar-based method of Daelemans, van den Bosch and Weijters and syllabification by analogy (SbA) from Marchand and Damper. In each case, performance is evaluated across the whole of each available corpus.

Syllabification performance is found to be very consistent across dictionaries in terms of the relative merits of the four techniques. The knowledge-based rule set performs poorly compared to the data-driven methods. Among the data-driven methods, SbA is easily the best. With regards to the dictionaries, best performance is obtained on the *Overlap* dictionary—probably because the overlap process removes idiosyncratic entries from *S&R* and *Wordsmyth*.

We believe there are sound reasons to expect the pattern of results seen here and the same trends showed on the problem of automatic pronunciation [19]. In our opinion, expert rule-based approaches suffer many drawbacks, including lack of conformance with real data, the limited ability of human experts to distinguish real from apparent regularities in very large datasets (like the effectively unbounded whole of natural language), and a tendency to over-rate dramatically the strength of weak, tentative linguistic theories.

6. Acknowledgements

This work was supported in part by funding from the Natural Sciences and Engineering Research Council of Canada (NSERC). In addition, the second author was funded by the National Research Council (NRC) Graduate Student Scholarship Supplement Program (GSSSP), and an Izaak Walton Killam Predoctoral Scholarship.

7. References

[1] E. Pulgram, *Syllable, Word, Nexus, Cursus*. The Hague, The Netherlands: Mouton, 1970.

[2] E. Selkirk, "The syllable," in *The Structure of Phonological Representations*, H. van der Hulst and N. Smith, Eds. Dordrecht, The Netherlands: Foris, 1982, vol. 2, pp. 337–383.

[3] K. J. Kohler, "Is the syllable a phonological universal?" *Journal of Linguistics*, vol. 2, no. ??, pp. 207–208, 1966.

[4] D. Crystal, *A First Dictionary of Linguistics and Phonetics*. London: André Deutsch, 1980.

[5] R. Treiman and A. Zukowski, "Toward an understanding of English syllabification," *Journal of Memory and Language*, vol. 29, no. 1, pp. 66–85, 1990.

[6] J. Goslin and U. H. Frauenfelder, "A comparison of theoretical and human syllabification," *Language and Speech*, vol. 44, no. 4, pp. 409–436, 2000.

[7] K. Müller, B. Möbius, and D. Prescher, "Inducing probabilistic syllable classes using multivariate clustering," in *Proceedings of 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong, China, 2000, pp. 225–232.

[8] D. Kahn, *Syllable-Based Generalizations in English Phonology*. Bloomington, IN: Indiana University Linguistics Club, 1976.

[9] G. N. Clements, "The role of the sonority cycle in core syllabification," 1988, working Papers of the Cornell Phonetics Laboratory, WPCPL No. 2, Research in Laboratory Phonology, Cornell University, Ithaca, NY.

[10] 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.

[11] 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.

[12] 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.

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

[14] R. I. Damper and Y. Marchand, "Information fusion approaches to the automatic pronunciation of print by analogy," *Information Fusion*, vol. 71, no. 2, pp. 207–220, 2006.

[15] Y. Marchand and R. I. Damper, "Can syllabification improve pronunciation by analogy?" *Natural Language Engineering*, vol. 13, no. 1, pp. 1–24, 2007.

[16] 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.

[17] W. Daelemans and A. van den Bosch, "Generalisation performance of backpropagation learning on a syllabification task," in *TWLT3: Connectionism and Natural Language Processing*, M. F. J. Drossaers and A. Nijholt, Eds. Enschede, The Netherlands: Twente University, 1992, pp. 27–37.

[18] W. Daelemans, A. van den Bosch, and T. Weijters, "IGTree: Using trees for compression and classification in lazy learning algorithms," *Artificial Intelligence Review*, vol. 11, no. 1–5, pp. 407–423, 1997.

[19] 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.