On the Syllabification of Phonemes

Susan Bartlett[†] and **Grzegorz Kondrak**[†] and **Colin Cherry**[‡]

[†]Department of Computing Science
University of Alberta
Edmonton, AB, T6G 2E8, Canada
{susan,kondrak}@cs.ualberta.ca

[‡]Microsoft Research
One Microsoft Way
Redmond, WA, 98052
colinc@microsoft.com

Abstract

Syllables play an important role in speech synthesis and recognition. We present several different approaches to the syllabification of phonemes. We investigate approaches based on linguistic theories of syllabification, as well as a discriminative learning technique that combines Support Vector Machine and Hidden Markov Model technologies. Our experiments on English, Dutch and German demonstrate that our transparent implementation of the sonority sequencing principle is more accurate than previous implementations, and that our language-independent SVM-based approach advances the current state-of-the-art, achieving word accuracy of over 98% in English and 99% in German and Dutch.

1 Introduction

Syllabification is the process of dividing a word into its constituent syllables. Although some work has been done on syllabifying orthographic forms (Müller et al., 2000; Bouma, 2002; Marchand and Damper, 2007; Bartlett et al., 2008), syllables are, technically speaking, phonological entities that can only be composed of strings of phonemes. Most linguists view syllables as an important unit of prosody because many phonological rules and constraints apply within syllables or at syllable boundaries (Blevins, 1995).

Apart from their purely linguistic significance, syllables play an important role in speech synthesis and recognition (Kiraz and Möbius, 1998; Pearson et al., 2000). The pronunciation of a given phoneme tends to vary depending on its location within a syllable. While actual implementations vary, text-to-speech (TTS) systems must have, at minimum, three components (Damper, 2001): a letter-to-phoneme (L2P) module, a prosody module, and a synthesis module. Syllabification can play a role in all three modules.

Because of the productive nature of language, a dictionary look-up process for syllabification is inadequate. No dictionary can ever contain all possible words in a language. For this reason, it is necessary to develop systems that can automatically syllabify out-of-dictionary words.

In this paper, we advance the state-of-the-art in both categorical (non-statistical) and supervised syllabification. We outline three categorical approaches based on common linguistic theories of syllabification. We demonstrate that when implemented carefully, such approaches can be very effective, approaching supervised performance. We also present a data-driven, discriminative solution: a Support Vector Machine Hidden Markov Model (SVM-HMM), which tags each phoneme with its syllabic role. Given enough data, the SVM-HMM achieves impressive accuracy thanks to its ability to capture context-dependent generalizations, while also memorizing inevitable exceptions. Our experiments on English, Dutch and German demonstrate that our SVM-HMM approach substantially outperforms the existing state-of-the-art learning approaches. Although direct comparisons are difficult, our system achieves over 99% word accuracy on German and Dutch, and the highest reported accuracy on English.

The paper is organized as follows. We outline common linguistic theories of syllabification in Section 2, and we survey previous computational sys-

Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL, pages 308–316,
Boulder, Colorado, June 2009.

tems in Section 3. Our linguistically-motivated approaches are described in Section 4. In Section 5, we describe our system based on the SVM-HMM. The experimental results are presented in Section 6.

2 Theories of Syllabification

There is some debate as to the exact structure of a syllable. However, phonologists are in general agreement that a syllable consists of a nucleus (vowel sound), preceded by an optional onset and followed by an optional coda. In many languages, both the onset and the coda can be complex, *i.e.*, composed of more than one consonant. For example, the word *breakfast* [brɛk-fəst] contains two syllables, of which the first has a complex onset [br], and the second a complex coda [st]. Languages differ with respect to various typological parameters, such as optionality of onsets, admissibility of codas, and the allowed complexity of the syllable constituents. For example, onsets are required in German, while Spanish prohibits complex codas.

There are a number of theories of syllabification; we present three of the most prevalent. The **Legality Principle** constrains the segments that can begin and end syllables to those that appear at the beginning and end of words. In other words, a syllable is not allowed to begin with a consonant cluster that is not found at the beginning of some word, or end with a cluster that is not found at the end of some word (Goslin and Frauenfelder, 2001). Thus, a word like *admit* [ədmɪt] must be syllabified [ədmɪt] because [dm] never appears word-initially or word-finally in English. A shortcoming of the legality principle is that it does not always imply a unique syllabification. For example, in a word like *askew* [əskju], the principle does not rule out any of [ə-skju], [əs-kju], or [əsk-ju], as all three employ legal onsets and codas.

The **Sonority Sequencing Principle** (SSP) provides a stricter definition of legality. The sonority of a sound is its inherent loudness, holding factors like pitch and duration constant (Crystal, 2003). Low vowels like [a], the most sonorous sounds, are high on the sonority scale, while plosive consonants like [t] are at the bottom. When syllabifying a word, SSP states that sonority should increase from the first phoneme of the onset to the syllable's nucleus, and then fall off to the coda (Selkirk, 1984). Consequently, in a word like *vintage* [vɪntɪʤ], we can rule out a syllabification like [vɪ-ntɪʤ] because [n] is more sonorant than [t]. However, SSP does not tell us whether to prefer [vɪn-tɪʤ] or [vɪnt-ɪʤ]. Moreover, when syllabifying a word like *vintner* [vɪntnər], the theory allows both [vɪn-tnər] and [vɪnt-nər], even though [tn] is an illegal onset in English.

Both the Legality Principle and SSP tell us which onsets and codas are permitted in legal syllables, and which are not. However, neither theory gives us any guidance when deciding between legal onsets. The **Maximal Onset Principle** addresses this by stating we should extend a syllable's onset at the expense of the preceding syllable's coda whenever it is legal to do so (Kahn, 1976). For example, the principle gives preference to [ə-skju] and [vɪn-tɪʤ] over their alternatives.

3 Previous Computational Approaches

Unlike tasks such as part of speech tagging or syntactic parsing, syllabification does not involve structural ambiguity. It is generally believed that syllable structure is usually predictable in a language provided that the rules have access to all conditioning factors: stress, morphological boundaries, part of speech, etymology, etc. (Blevins, 1995). However, in speech applications, the phonemic transcription of a word is often the only linguistic information available to the system. This is the common assumption underlying a number of computational approaches that have been proposed for the syllabification of phonemes.

Daelemans and van den Bosch (1992) present one of the earliest systems on automatic syllabification: a neural network-based implementation for Dutch. Daelemans et al. (1997) also explore the application of exemplar-based generalization (EBG), sometimes called instance-based learning. EBG generally performs a simple database look-up to syllabify a test pattern, choosing the most common syllabification. In cases where the test pattern is not found in the database, the most similar pattern is used to syllabify the test pattern.

Hidden Markov Models (HMMs) are another popular approach to syllabification. Krenn (1997) introduces the idea of treating syllabification as a

tagging task. Working from a list of syllabified phoneme strings, she automatically generates tags for each phone. She uses a second-order HMM to predict sequences of tags; syllable boundaries can be trivially recovered from the tags. Demberg (2006) applies a fourth-order HMM to the syllabification task, as a component of a larger German text-to-speech system. Schmid et al. (2007) improve on Demberg's results by applying a fifth-order HMM that conditions on both the previous tags and their corresponding phonemes.

Kiraz and Möbius (1998) present a weighted finite-state-based approach to syllabification. Their language-independent method builds an automaton for each of onsets, nuclei, and codas, by counting occurrences in training data. These automatons are then composed into a transducer accepting sequences of one or more syllables. They do not report quantitative results for their method.

Pearson et al. (2000) compare two rule-based systems (they do not elaborate on the rules employed) with a CART decision tree-based approach and a "global statistics" algorithm. The global statistics method is based on counts of consonant clusters in contexts such as word boundaries, short vowels, or long vowels. Each test word has syllable boundaries placed according to the most likely location given a cluster and its context. In experiments performed with their in-house dataset, their statistics-based method outperforms the decision-tree approach and the two rule-based methods.

Müller (2001) presents a hybrid of a categorical and data-driven approach. First, she manually constructs a context-free grammar of possible syllables. This grammar is then made probabilistic using counts obtained from training data. Müller (2006) attempts to make her method language-independent. Rather than hand-crafting her context-free grammar, she automatically generates all possible onsets, nuclei, and codas, based on the phonemes existing in the language. The results are somewhat lower than in (Müller, 2001), but the approach can be more easily ported across languages.

Goldwater and Johnson (2005) also explore using EM to learn the structure of English and German phonemes in an unsupervised setting, following Müller in modeling syllable structure with PCFGs. They initialize their parameters using a deterministic parser implementing the sonority principle and estimate the parameters for their maximum likelihood approach using EM.

Marchand et al. (2007) apply their Syllabification by Analogy (SbA) technique, originally developed for orthographic forms, to the pronunciation domain. For each input word, SbA finds the most similar substrings in a lexicon of syllabified phoneme strings and then applies the dictionary syllabifications to the input word. Their survey paper also includes comparisons with a method broadly based on the legality principle. The authors find their legality-based implementation fares significantly worse than SbA.

4 Categorical Approaches

Categorical approaches to syllabification are appealing because they are efficient and linguistically intuitive. In addition, they require little or no syllable-annotated data. We present three *categorical* algorithms that implement the linguistic insights outlined in Section 2. All three can be viewed as variations on the basic pseudo-code shown in Figure 1. Every vowel is labeled as a nucleus, and every consonant is labeled as either an onset or a coda. The algorithm labels all consonants as onsets unless it is illegal to do so. Given the labels, it is straightforward to syllabify a word. The three methods differ in how they determine a "legal" onset.

As a rough baseline, the MAXONSET implementation considers all combinations of consonants to be legal onsets. Only word-final consonants are labeled as codas.

LEGALITY combines the Legality Principle with onset maximization. In our implementation, we collect all word-initial consonant clusters from the corpus and deem them to be legal onsets. With this method, no syllable can have an onset that does not appear word-initially in the training data. We do not test for the legality of codas. The performance of LEGALITY depends on the number of phonetic transcriptions that are available, but the transcriptions need not be annotated with syllable breaks.

SONORITY combines the Sonority Sequencing Principle with onset maximization. In this approach, an onset is considered legal if every member of the onset ranks lower on the sonority scale than ensuing

```
until current phoneme is a vowel
    label current phoneme as an onset
end loop
until all phonemes have been labeled
    label current phoneme as a nucleus
    if there are no more vowels in the word
        label all remaining consonants as codas
    else
        onset := all consonants before next vowel
        coda := empty
        until onset is legal
            coda := coda plus first phoneme of onset
            onset := onset less first phoneme
        end loop
    end if
end loop
Insert syllable boundaries before onsets
```

Figure 1: Pseudo-code for syllabifying a string of phonemes.

consonants. SONORITY requires no training data because it implements a sound linguistic theory. However, an existing development set for a given language can help with defining and validating additional language-specific constraints.

Several sonority scales of varying complexity have been proposed. For example, Selkirk (1984) specifies a hierarchy of eleven distinct levels. We adopt a minimalistic scale shown in Figure 2. which avoids most of the disputed sonority contrasts (Jany et al., 2007). We set the sonority distance parameter to 2, which ensures that adjacent consonants in the onset differ by at least two levels of the scale. For example, [pr] is an acceptable onset because it is composed of an obstruent and a liquid, but [pn] is not, because nasals directly follow obstruents on our sonority scale.

In addition, we incorporate several English-specific constraints listed by Kenstowicz (1994, pages 257–258). The constraints, or *filters*, prohibit complex onsets containing:

(i) two labials (e.g., [pw], [bw], [fw], [vw]),

(ii) a non-strident coronal followed by a lateral (e.g., [tl], [dl], [θl])

(iii) a voiced fricative (e.g., [vr], [zw], except [vj]),

(iv) a palatal consonant (e.g., [ʃl], [ʧr], except [ʃr]).

Sound	**Examples**	**Level**
Vowels	u, ə, ...	4
Glides	w, j, ...	3
Liquids	l, r, ...	2
Nasals	m, ŋ, ...	1
Obstruents	g, θ, ...	0

Figure 2: The sonority scale employed by SONORITY.

A special provision allows for prepending the phoneme [s] to onsets beginning with a voiceless plosive. This reflects the special status of [s] in English, where onsets like [sk] and [sp] are legal even though the sonority is not strictly increasing.

5 Supervised Approach: SVM-HMM

If annotated data is available, a classifier can be trained to predict the syllable breaks. A Support Vector Machine (SVM) is a discriminative supervised learning technique that allows for a rich feature representation of the input space. In principle, we could use a multi-class SVM to classify each phoneme according to its position in a syllable on the basis of a set of features. However, a traditional SVM would treat each phoneme in a word as an independent instance, preventing us from considering interactions between labels. In order to overcome this shortcoming, we employ an SVM-HMM[1] (Altun et al., 2003), an instance of the Structured SVM formalism (Tsochantaridis et al., 2004) that has been specialized for sequence tagging.

When training a structured SVM, each training instance x_i is paired with its label y_i, drawn from the set of possible labels, Y_i. In our case, the training instances x_i are words, represented as sequences of phonemes, and their labels y_i are syllabifications, represented as sequences of onset/nucleus/coda tags. For each training example, a feature vector $\Psi(x, y)$ represents the relationship between the example and a candidate tag sequence. The SVM finds a weight vector w, such that $w \cdot \Psi(x, y)$ separates correct taggings from incorrect taggings by as large a margin as possible. Hamming distance D_H is used to capture how close a wrong sequence y is to y_i, which

[1] http://svmlight.joachims.org/svm_struct.html

in turn impacts the required margin. Tag sequences that share fewer tags in common with the correct sequence are separated by a larger margin.

Mathematically, a (simplified) statement of the SVM learning objective is:

$$\forall_i \forall_{y \in Y_i, y \neq y_i} : \\ [\Psi(x_i, y_i) \cdot w > \Psi(x_i, y) \cdot w + D_H(y, y_i)] \quad (1)$$

This objective is only satisfied when w tags all training examples correctly. In practice, slack variables are introduced, which allow us to trade off training accuracy and the complexity of w via a cost parameter. We tune this parameter on our development set.

The SVM-HMM training procedure repeatedly uses the Viterbi algorithm to find, for the current w and each (x_i, y_i) training pair, the sequence y that most drastically violates the inequality shown in Equation 1. These incorrect tag sequences are added to a growing set, which constrains the quadratic optimization procedure used to find the next w. The process iterates until no new violating sequences are found, producing an approximation to the inequality over all $y \in Y_i$. A complete explanation is given by Tsochantaridis et al. (2004).

Given a weight vector w, a structured SVM tags new instances x according to:

$$\text{argmax}_{y \in Y} [\Psi(x, y) \cdot w] \quad (2)$$

The SVM-HMM gets the HMM portion of its name from its use of the HMM Viterbi algorithm to solve this argmax.

5.1 Features

We investigated several tagging schemes, described in detail by Bartlett (2007). During development, we found that tagging each phoneme with its syllabic role (Krenn, 1997) works better than the simple binary distinction between syllable-final and other phonemes (van den Bosch, 1997). We also discovered that accuracy can be improved by numbering the tags. Therefore, in our tagging scheme, the single-syllable word *strengths* [strɛŋθs] would be labeled with the sequence {O1 O2 O3 N1 C1 C2 C3}.

Through the use of the Viterbi algorithm, our feature vector $\Psi(x, y)$ is naturally divided into emission and transition features. Emission features link an aspect of the input word x with a single tag in the sequence y. Unlike a generative HMM, these emission features do not require any conditional independence assumptions. Transition features link tags to tags. Our only transition features are counts of adjacent tag pairs occurring in y.

For the emission features, we use the current phoneme and a fixed-size context window of surrounding phonemes. Thus, the features for the phoneme [k] in *hockey* [hɑki] might include the [ɑ] preceding it, and the [i] following it. In experiments on our development set, we found that the optimal window size is nine: four phonemes on either side of the focus phoneme. Because the SVM-HMM is a linear classifier, we need to explicitly state any important conjunctions of features. This allows us to capture more complex patterns in the language that unigrams alone cannot describe. For example, the bigram [ps] is illegal as an onset in English, but perfectly reasonable as a coda. Experiments on the development set showed that performance peaked using all unigrams, bigrams, trigrams, and four-grams found within our context window.

Method	English
MAXONSET	61.38
LEGALITY	93.16
SONORITY	95.00
SVM-HMM	98.86
tsylb	93.72

Table 1: Word accuracy on the CELEX dataset.

6 Syllabification Experiments

We developed our approach using the English portion of the CELEX lexical database (Baayen et al., 1995). CELEX provides the phonemes of a word and its correct syllabification. It does not designate the phonemes as onsets, nuclei, or codas, which is the labeling we want to predict. Fortunately, extracting the labels from a syllabified word is straightforward. All vowel phones are assigned to be nuclei; consonants preceding the nucleus in a syllable are assigned to be onsets, while consonants following the nucleus in a syllable are assigned to be codas.

The results in Table 1 were obtained on a test set of 5K randomly selected words. For training the SVM-HMM, we randomly selected 30K words not

appearing in the test set, while 6K training examples were held out for development testing. We report the performance in terms of word accuracy (entire words syllabified correctly). Among the categorical approaches, SONORITY clearly outperforms not only LEGALITY, but also *tsylb* (Fisher, 1996), an implementation of the complex algorithm of Kahn (1976), which makes use of lists of legal English onsets. Overall, our SVM-based approach is a clear winner.

The results of our discriminative method compares favorably with the results of competing approaches on English CELEX. Since there are no standard train-test splits for syllabification, the comparison is necessarily indirect, but note that our training set is substantially smaller. For her language-independent PCFG-based approach, Müller (2006) reports 92.64% word accuracy on the set of 64K examples from CELEX using 10-fold cross-validation. The Learned EBG approach of van den Bosch (1997) achieves 97.78% word accuracy when training on approximately 60K examples. Therefore, our results represent a nearly 50% reduction of the error rate.

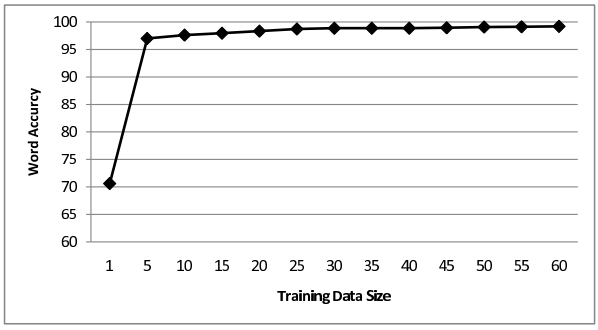

Figure 3: Word accuracy on English CELEX as a function of the number of thousands of training examples.

Though the SVM-HMM's training data requirements are lower than previous supervised syllabification approaches, they are still substantial. Figure 3 shows a learning curve over varying amounts of training data. Performance does not reach acceptable levels until 5K training examples are provided.

6.1 Error Analysis

There is a fair amount of overlap in the errors made by the SVM-HMM and the SONORITY. Table 4 shows a few characteristic examples. The CELEX syllabifications of *tooth-ache* and *pass-ports* follow the morphological boundaries of the compound words. Morphological factors are a source of errors for both approaches, but significantly more so for SONORITY. The performance difference comes mainly from the SVM's ability to handle many of these morphological exceptions. The SVM generates the correct syllabification of *northeast* [norθ-ist], even though an onset of [θ] is perfectly legal. On the other hand, the SVM sometimes overgeneralizes, as in the last example in Table 4.

SVM-HMM	SONORITY	
tu-θek	tu-θek	*toothache*
pae-sports	pae-sports	*passports*
norθ-ist	nor-θist	*northeast*
dɪs-plizd	dɪ-splizd	*displeased*
dɪs-koz	**dɪ-skoz**	*discos*

Figure 4: Examples of syllabification errors. (Correct syllabifications are shown in bold.)

6.2 The NETtalk Dataset

Marchand et al. (2007) report a disappointing word accuracy of 54.14% for their legality-based implementation, which does not accord with the results of our categorical approaches on English CELEX. Consequently, we also apply our methods to the dataset they used for their experiments: the NETtalk dictionary. NETtalk contains 20K English words; in the experiments reported here, we use 13K training examples and 7K test words.

As is apparent from Table 2, our performance degrades significantly when switching to NETtalk. The steep decline found in the categorical methods is particularly notable, and indicates significant divergence between the syllabifications employed in the two datasets. Phonologists do not always agree on the correct syllable breaks for a word, but the NETtalk syllabifications are often at odds with linguistic intuitions. We randomly selected 50 words and compared their syllabifications against those found in Merriam-Webster Online. We found that CELEX syllabifications agree with Merriam-Webster in 84% of cases, while NETtalk only agrees 52% of the time.

Figure 5 shows several words from the NETtalk

Method	English
MAXONSET	33.64
SONORITY	52.80
LEGALITY	53.08
SVM-HMM	92.99

Table 2: Word accuracy on the NETtalk dataset.

and CELEX datasets. We see that CELEX follows the maximal onset principle consistently, while NETtalk does in some instances but not others. We also note that there are a number of NETtalk syllabifications that are clearly wrong, such as the last two examples in Figure 5. The variability of NETtalk is much more difficult to capture with any kind of principled approach. Thus, we argue that low performance on NETtalk indicate inconsistent syllabifications within that dataset, rather than any actual deficiency of the methods.

NETtalk	**CELEX**	
ʧaes-taɪz	ʧae-staɪz	*chastise*
rɛz-ɪd-ənz	rɛ-zɪ-dənz	*residence*
dɪ-strɔɪ	dɪ-strɔɪ	*destroy*
fo-tɑn	fo-tɑn	*photon*
ɑr-pɛʤ-io	ɑr-pɛ-ʤi-o	*arpeggio*
ðer-ə-baʊ-t	ðɛ-rə-baʊt	*thereabout*

Figure 5: Examples of CELEX and NETtalk syllabifications.

NETtalk's variable syllabification practices notwithstanding, the SVM-HMM approach still outperforms the previous benchmark on the dataset. Marchand et al. (2007) report 88.53% word accuracy for their SbA technique using leave-one-out testing on the entire NETtalk set (20K words). With fewer training examples, we reduce the error rate by almost 40%.

6.3 Other Languages

We performed experiments on German and Dutch, the two other languages available in the CELEX lexical database. The German and Dutch lexicons of CELEX are larger than the English lexicon. For both languages, we selected a 25K test set, and two different training sets, one containing 50K words and the other containing 250K words. The results are presented in Table 3.

Method	German	Dutch
MAXONSET	19.51	23.44
SONORITY	76.32	77.51
LEGALITY	79.55	64.31
SVM-HMM (50K words)	99.26	97.79
SVM-HMM (250K words)	99.87	99.16

Table 3: Word accuracy on the CELEX dataset.

While our SVM-HMM approach is entirely language independent, the same cannot be said about other methods. The maximal onset principle appears to hold much more strongly for English than for German and Dutch (e.g., *patron*: [pe-trən] vs. [pat-ron]). LEGALITY and SONORITY also appear to be less effective, possibly because of greater tendency for syllabifications to match morphological boundaries (e.g., English *exclusive*: [ɪk-sklu-sɪv] vs. Dutch *exclusief* [ɛks-kly-zif]). SONORITY is further affected by our decision to employ the constraints of Kenstowicz (1994), although they clearly pertain to English. We expect that adapting them to specific languages would bring the results closer to the level of the English experiments.

Although our SVM system is tuned using an English development set, the results on both German and Dutch are excellent. We could not find any quantitative data for comparisons on Dutch, but the comparison with the previously reported results on German CELEX demonstrates the quality of our approach. The numbers that follow refer to 10-fold cross-validation on the entire lexicon (over 320K entries) unless noted otherwise. Krenn (1997) obtains *tag* accuracy of 98.34%, compared to our system's tag accuracy of 99.97% when trained on 250K words. With a hand-crafted grammar, Müller (2002) achieves 96.88% word accuracy on CELEX-derived syllabifications, with a training corpus of two million tokens. Without a hand-crafted grammar, she reports 90.45% word accuracy (Müller, 2006). Applying a standard smoothing algorithm and fourth-order HMM, Demberg (2006) scores 98.47% word accuracy. A fifth-order joint N-gram model of Schmid et al. (2007) achieves 99.85% word accuracy with about 278K training points. However, unlike generative approaches, our

Method	English	German
SONORITY	97.0	94.2
SVM-HMM	99.9	99.4
Categorical Parser	94.9	92.7
Maximum Likelihood	98.1	97.4

Table 4: Word accuracy on the datasets of Goldwater and Johnson (2005).

SVM-HMM can condition each emission on large portions of the input using only a first-order Markov model, which implies much faster syllabification performance.

6.4 Direct Comparison with an MLE approach

The results of the competitive approaches that have been quoted so far (with the exception of *tsylb*) are not directly comparable, because neither the respective implementations, nor the actual train-test splits are publicly available. However, we managed to obtain the English and German data sets used by Goldwater and Johnson (2005) in their study, which focused primarily on unsupervised syllabification. Their experimental framework is similar to (Müller, 2001). They collect words from running text and create a training set of 20K tokens and a test set of 10K tokens. The running text was taken from the Penn WSJ and ECI corpora, and the syllabified phonemic transcriptions were obtained from CELEX. Table 4 compares our experimental results with their reported results obtained with: (a) supervised Maximum Likelihood training procedures, and (b) a Categorical Syllable Parser implementing the principles of sonority sequencing and onset maximization without Kenstowicz's (1994) onset constraints.

The accuracy figures in Table 4 are noticeably higher than in Table 1. This stems from fundamental differences in the experimental set-up; Goldwater and Johnson (2005) test on tokens as found in text, therefore many frequent short words are duplicated. Furthermore, some words occur during both training and testing, to the benefit of the supervised systems (SVM-HMM and Maximum Likelihood). Nevertheless, the results confirm the level of improvement obtained by both our categorical and supervised approaches.

7 Conclusion

We have presented several different approaches to the syllabification of phonemes. The results of our linguistically-motivated algorithms, show that it is possible to achieve adequate syllabification word accuracy in English with no little or no syllable-annotated training data. We have demonstrated that the poor performance of categorical methods on English NETtalk actually points to problems with the NETtalk annotations, rather than with the methods themselves.

We have also shown that SVM-HMMs can be used to great effect when syllabifying phonemes. In addition to being both efficient and language-independent, they establish a new state-of-the-art for English and Dutch syllabification. However, they do require thousands of labeled training examples to achieve this level of accuracy. In the future, we plan to explore a hybrid approach, which would benefit from both the generality of linguistic principles and the smooth exception-handling of supervised techniques, in order to make best use of whatever data is available.

Acknowledgements

We are grateful to Sharon Goldwater for providing the experimental data sets for comparison. This research was supported by the Natural Sciences and Engineering Research Council of Canada and the Alberta Informatics Circle of Research Excellence.

References

Yasemin Altun, Ioannis Tsochantaridis, and Thomas Hofmann. 2003. Hidden markov support vector machines. *Proceedings of the 20Th International Conference on Machine Learning (ICML)*.

R. Baayen, R. Piepenbrock, and L. Gulikers. 1995. The CELEX lexical database (CD-ROM).

Susan Bartlett, Grzegorz Kondrak, and Colin Cherry. 2008. Automatic syllabification with structured SVMs for letter-to-phoneme conversion. In *Proceedings of ACL-08: HLT*, pages 568–576, Columbus, Ohio.

Susan Bartlett. 2007. Discriminative approach to automatic syllabification. Master's thesis, Department of Computing Science, University of Alberta.

Juliette Blevins. 1995. The syllable in phonological theory. In John Goldsmith, editor, *The handbook of phonological theory*, pages 206–244. Blackwell.

Gosse Bouma. 2002. Finite state methods for hyphenation. *Natural Language Engineering*, 1:1–16.

David Crystal. 2003. *A dictionary of linguistics and phonetics*. Blackwell.

Walter Daelemans and Antal van den Bosch. 1992. Generalization performance of backpropagaion learning on a syllabification task. In *Proceedings of the 3rd Twente Workshop on Language Technology*, pages 27–38.

Walter Daelemans, Antal van den Bosch, and Ton Weijters. 1997. IGTree: Using trees for compression and classification in lazy learning algorithms. *Artificial Intelligence Review*, pages 407–423.

Robert Damper. 2001. Learning about speech from data: Beyond NETtalk. In *Data-Driven Techniques in Speech Synthesis*, pages 1–25. Kluwer Academic Publishers.

Vera Demberg. 2006. Letter-to-phoneme conversion for a German text-to-speech system. Master's thesis, University of Stuttgart.

William Fisher. 1996. Tsylb syllabification package. ftp://jaguar.ncsl.nist.gov/pub/tsylb2-1.1.tar.Z. Last accessed 31 March 2008.

Sharon Goldwater and Mark Johnson. 2005. Representational bias in usupervised learning of syllable structure. In *Prcoeedings of the 9th Conference on Computational Natural Language Learning (CoNLL)*, pages 112–119.

Jeremy Goslin and Ulrich Frauenfelder. 2001. A comparison of theoretical and human syllabification. *Language and Speech*, 44:409–436.

Carmen Jany, Matthew Gordon, Carlos M Nash, and Nobutaka Takara. 2007. How universal is the sonority hierarchy? A cross-linguistic study. In *16th International Congress of Phonetic Sciences*, pages 1401–1404.

Daniel Kahn. 1976. *Syllable-based generalizations in English Phonology*. Ph.D. thesis, Indiana University.

Michael Kenstowicz. 1994. *Phonology in Generative Grammar*. Blackwell.

George Kiraz and Bernd Möbius. 1998. Multilingual syllabification using weighted finite-state transducers. In *Proceedings of the 3rd Workshop on Speech Synthesis*.

Brigitte Krenn. 1997. Tagging syllables. In *Proceedings of Eurospeech*, pages 991–994.

Yannick Marchand and Robert Damper. 2007. Can syllabification improve pronunciation by analogy of English? *Natural Language Engineering*, 13(1):1–24.

Yannick Marchand, Connie Adsett, and Robert Damper. 2007. Automatic syllabification in English: A comparison of different algorithms. *Language and Speech*. To appear.

Karin Müller, Bernd Möbius, and Detlef Prescher. 2000. Inducing probabilistic syllable classes using multivariate clustering. In *Prcoeedings of the 38th meeting of the ACL*.

Karin Müller. 2001. Automatic detection of syllable boundaries combining the advantages of treebank and bracketed corpora training. *Proceedings on the 39Th Meeting of the ACL*.

Karin Müller. 2002. Probabilistic context-free grammars for phonology. *Proceedings of the 6th Workshop of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 80–90.

Karin Müller. 2006. Improving syllabification models with phonotactic knowledge. *Proceedings of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology At HLT-NAACL*.

Steve Pearson, Roland Kuhn, Steven Fincke, and Nick Kibre. 2000. Automatic methods for lexical stress assignment and syllabification. In *Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP)*.

Helmut Schmid, Bernd Möbius, and Julia Weidenkaff. 2007. Tagging syllable boundaries with joint N-gram models. In *Proceedings of Interspeech*.

Elisabeth Selkirk. 1984. On the major class features and syllable theory. In *Language Sound Structure*. MIT Press.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. *Proceedings of the 21st International Conference on Machine Learning (ICML)*.

Antal van den Bosch. 1997. *Learning to pronounce written words: a study in inductive language learning*. Ph.D. thesis, Universiteit Maastricht.