PBML

The Prague Bulletin of Mathematical Linguistics

NUMBER 100 OCTOBER 2013 113–131

Sequence Segmentation by Enumeration: An Exploration

Steffen Eger

Goethe University
Frankfurt am Main/Germany

Abstract

We investigate *exhaustive enumeration* and subsequent *language model evaluation* (E&E approach) as an alternative to solving the sequence segmentation problem. We show that, under certain conditions (on string lengths and regarding a possibility to accurately estimate the number of segments), which are satisfied for important NLP applications, such as phonological segmentation, syllabification, and morphological segmentation, the E&E approach is feasible and promises superior results than the standard sequence labeling approach to sequence segmentation.

1. Introduction

By *sequence segmentation*, we mean the splitting of a sequence $x = x_1 \dots x_n$ consisting of n characters, each from an alphabet Σ, into non-overlapping *segments*, or *parts*, such that the concatenation of the segments, in the 'original' order, precisely yields x. Usually, in applications, we do not seek an arbitrary segmentation of x but the 'most suitable', where suitability may be defined, particularly in a supervised setting as we consider, with respect to a given distribution of data. In NLP, segmentations of sequences may occur in a variety of contexts such as *morphological segmentation*, the breaking of words into morphemes, *syllabification*, the breaking of words into syllables, *phonological segmentation*, the breaking of words into 'phonological units', or *word segmentation* (cf. Goldwater et al., 2009), the breaking of sentences into words. For example, the sequence $x =$ *phoenix* may admit suitable segmentations as in

ph-oe-n-i-x   phoe-nix   phoenix

for phonological segmentation, syllabification, and morphological segmentation, respectively, and where we delineate segments in an intuitive manner.

Corresponding author: eger.steffen@gmail.com

In a supervised learning context, sequence segmentation may be considered a *sequence labeling problem* where the labels indicate whether or not a split occurs at a given character position. For instance, the above segmentations of x = *phoenix* may be encoded as

```
p h o e n i x   p h o e n i x   p h o e n i x
0 0 1 0 1 1 1   0 0 0 0 1 0 0   0 0 0 0 0 0 0
```

where a '1' indicates a split.

Alternatively, we may view sequence segmentation as an 'evaluation' problem, in an apparently intuitive manner. Namely, given a test string x, enumerate all possible segmentations of x and evaluate, or score, each of them using a language model trained on the training data. Such an approach is potentially superior because it allows to take a 'word' (rather than 'character') perspective on the data. Moreover and most importantly, exhaustive search for evaluation is an *exact* paradigm for *arbitrary* evaluation models, whereas sequence labeling models typically make crucial, e.g., independence assumptions on scoring functions (see our discussion in Section 5.4).

The problem with the 'evaluation viewpoint', and the exhaustive search it naïvely relies upon, is that there are 2^{n-1} possible segmentations of a sequence x of length n, i.e., the search space is exponential in the number of characters of x, which makes the approach apparently impractical for all but very short sequences. In the current work, we challenge this claim. We present a simple model for sequence segmentation that rests on the evaluation viewpoint outlined above and on exhaustive enumeration, and that works well, as we demonstrate, under the following two conditions,

- for a given test string x, the *number* of segments of an optimal segmentation of x is known (or known to be in a 'small' interval) or can easily and accurately be predicted,
- for a given test string x, the length n of x is not 'too large' (e.g., is certainly less than 50) and/or the possible lengths of segments are not 'too large' (e.g., are less than 10 or so).

As we show in the next sections, when these assumptions are satisfied, exhaustive enumeration is in fact cheap and can easily be implemented. Consequently, in this situation, it is unproblematic to apply the evaluation viewpoint to sequence segmentation, which, as we show via experiments, may yield superior results for the sequence segmentation problem; we indicate error rate decreases between 5 and 42% over state-of-the-art sequence labeling approaches across different data sets. In the current work, we demonstrate, moreover, that our two criteria outlined above apparently hold for a number of 'string related' sequence segmentation problems in NLP such as morphological segmentation, syllabification, and phonological segmentation (they certainly do *not* apply to, e.g., word segmentation). In this respect, hence, our methodology is apparently well suited to a class of important NLP applications.

This work is structured as follows. In Section 2, we more thoroughly investigate the search space for (naïve) sequence segmentation. We do so by referring to results on *restricted integer compositions*, a field in mathematical combinatorics that has recently gained increasing interest (Heubach and Mansour, 2004; Bender and Canfield, 2005; Shapcott, 2012; Eger, 2013). In Section 3, we illustrate our approach in more detail, before describing our data in Section 4. Then, in Section 5, we detail our experiments on sequence segmentation. Since our approach may be prone to misinterpretation, we discuss and summarize the intentions of our approach and the lessons that can be learned from it in Section 5.4. In Section 6, we discuss related work and in Section 7, we conclude.

2. Search Space for Sequence Segmentation

We first define integer compositions and then show their relationship to sequence segmentations.

Let $n, k \in \mathbb{N} = \{0, 1, 2 \dots\}$. An *integer composition of* n *with* k *parts* is a k-tuple $(\pi_1, \dots, \pi_k) \in \mathbb{N}^k$ such that $\pi_1 + \cdots + \pi_k = n$. Denote by $\mathcal{C}(n, k)$ the set of all integer compositions of n with k parts. Obviously, there exists a 'natural' bijection between segmentations of a sequence $\mathbf{x} = x_1 \dots x_n$ of length n with k segments and integer compositions of n with k parts in which the sizes of parts correspond to the lengths of the respective segments as in

$$\begin{array}{ccccccccccc} & \text{ph} & & \text{oe} & & \text{n} & & \text{i} & & \text{x} \\ 7 = & 2 & + & 2 & + & 1 & + & 1 & + & 1 \end{array}$$

Thus, the number of sequence segmentations of $\mathbf{x} = x_1 \dots x_n$ with k segments equals the number of integer compositions of n with k parts, $|\mathcal{C}(n, k)| = |\mathcal{S}(n, k)|$, where $\mathcal{S}(n, k)$ denotes the set of all segmentations of $x_1 \dots x_n$ with k segments. There are several well-known combinatorial results regarding the number of integer compositions of n with k parts. For example,

$$|\mathcal{C}(n, k)| = \binom{n-1}{k-1},$$

where $\binom{n}{k}$ denotes the respective binomial coefficient. Moreover, less well-known, the number of *restricted integer compositions*, that is, where each part is restricted to lie within an interval $A = \{\xi_{\min}, \xi_{\min} + 1, \dots, \xi_{\max}\}$, $\xi_{\min}, \xi_{\max} \in \mathbb{N}$, with $\xi_{\min} \leq \xi_{\max}$, is given by the *extended binomial coefficient* (Fahssi, 2012; Eger, 2013)[1]

$$|\mathcal{C}_A(n, k)| = \binom{k}{n - \xi_{\min} k}_{\xi_{\max} - \xi_{\min} + 1}, \tag{1}$$

where $\binom{k}{n}_{l+1}$ arises as the coefficient of X^n of the polynomial $(1 + X + X^2 + \dots + X^l)^k$ and where we denote by $\mathcal{C}_A(n, k)$ the set of all compositions of n with k parts, each

[1]Extended binomial coefficients share many interesting properties with ordinary binomial coefficients; see the discussions in the cited works.

within the interval A. As above, it obviously holds that $|\mathcal{C}_A(n,k)| = |\mathcal{S}_A(n,k)|$, where $\mathcal{S}_A(n,k)$ is the set of all segmentations of a sequence of length n with k segments where segment lengths are restricted to lie within A. Restrictions on segment lengths may be useful and justified in NLP applications; for instance, in phonological segmentation, we would hardly expect a segment to exceed, say, length 4,[2] and in syllabification, syllables that exceed, say, length 9 or 10, are presumably very rare across different languages.

As concerns the total number of sequence segmentations of a sequence of length n, we have

$$|\mathcal{S}(n)| = |\mathcal{C}(n)| = \sum_{k\geq 1} \binom{n-1}{k-1} = 2^{n-1},$$

where we use analogous notation as above. For restricted sequence segmentations, closed form formulas are more difficult to obtain. For $A = \{1, \ldots, b\}$, $|\mathcal{S}_A(n)|$ is a *generalized Fibonacci number* satisfying the recurrence

$$\left|\mathcal{S}_{\{1,\ldots,b\}}(n)\right| = \sum_{i=1}^{b} \left|\mathcal{S}_{\{1,\ldots,b\}}(n-i)\right|.$$

Asymptotic formulas are given, e.g., by Malandro (2011) as

$$\left|\mathcal{S}_{\{1,\ldots,b\}}(n)\right| \sim \frac{\phi^{n+1}}{G'(1/\phi)}, \tag{2}$$

where ϕ is the unique positive real solution to $\sum_{i=1}^{b} X^{-i} = 1$ and where $G(X) = \sum_{i=1}^{b} X^{i}$ and G' denotes its first derivative. For instance, there are 5 restricted segmentations of $x = x_1x_2x_3x_4$ where $A = \{1,2\}$ — namely, $x_1x_2 - x_3x_4$; $x_1x_2 - x_3 - x_4$; $x_1 - x_2x_3 - x_4$; $x_1 - x_2 - x_3x_4$; and $x_1 - x_2 - x_3 - x_4$ — while the approximation formula gives the (very close) value 4.96 for $\left|\mathcal{S}_{\{1,2\}}(4)\right|$, since $\phi = (1+\sqrt{5})/2$ in this case. Formula (2) also indicates that the number of segmentations of a sequence x asymptotically grows *exponentially* in the length n of x, even under restrictions on segment sizes, although, for any given n, there might be much fewer restricted segmentations than in the unrestricted case. For example, $\left|\mathcal{S}_{\{1,2\}}(15)\right| = 987$, while $|\mathcal{S}(15)| = 2^{14} = 16384$.

Efficient algorithms for generating restricted integer compositions have recently been suggested in Opdyke (2010); Page (2012) and a Matlab implementation of the algorithm designed in Opdyke (2010) is available from `http://www.mathworks.com/matlabcentral/fileexchange/27110-restricted-integer-composition`.

[2]See Table 1 for examples.

3. Method

As we have indicated, our approach for (supervised) sequence segmentation is as follows. Given labeled data (i.e., with 'gold standard' segmentations), at *training time*, we simply train a language model LM on the training data set. At *test time*, we predict the segmentation of a test string x by exhaustively enumerating all possible segmentations of x and evaluating each of them via LM. The best scoring segmentation is then our prediction for x. We refer to this approach as E&E (for 'enumerate and evaluate'). As mentioned, since enumerating (really) *all* possible segmentations of x is generally impracticable (even for restricted segmentations), we crucially rely on a 'number of parts' prediction model PM; predicting the number of parts of the correct segmentation of x is a simpler problem than actually providing the correct segmentation. We outline a possible strategy for specifying PM below.

We consider both a *word level*, LM-W, and a *character level*, LM-C, language model for our E&E approach. The character level model views (training) strings as a sequence of 'characters' as in *ph-oe-n-ix* (including the split information) while the word level model views the same strings as a sequence of 'words' as in *ph oe n i x* (which also includes the split information). Intuitively, we would expect both models to perform differently in different situations. For example, in syllabification, segmentations crucially depend on character information (e.g., whether or not the current character is a vowel or a consonant) while in word segmentation or morphological segmentation, a word level view may be a 'superior' perspective.

4. Data and Its Statistical Properties

We use CELEX (Baayen et al., 1996) as our lexical database. CELEX provides information on orthographical (syllabification) and morphological segmentation for German, English, and Dutch. Moreover, it provides phonological transcriptions for the three languages. To generate phonological segmentations from these, we first align words with their phonological representations via a monotone many-to-many aligner (cf. Eger, 2012) and then retrieve the phonologically segmented words. For the phonology data, we use random subsets of data from the Pascal challenge (Van den Bosch et al., 2006), which, in the case of German and Dutch, is directly based on CELEX but already provides a filtering; here, we also include data on French from the Pascal challenge, which is based on the Brulex database (Content et al., 1990). In the case of orthographical and morphological segmentation, we remove all duplicates and multi-word entries from CELEX and focus on random subsets of given sizes, as indicated in Table 2. In Table 1, we give examples of gold standard segmented data, across the different languages and segmentation domains.

Table 2 summarizes statistical properties of our data sets. The first three columns refer to the minimum, maximum, and average number of parts of segmentations in the various gold standard alignments. The next three columns refer to the minimum,

G-P	b-e-r-ei-t, sch-uh, sch-n-ee-m-a-tsch, s-ä-tt-i-g-u-ng
E-P	ear-th-en, th-r-ough, o-ff-sh-oo-t, a-gg-r-e-ss-i-ve
D-P	sj-o-tt-en, w-ij-n-h-ui-s, i-mm-uu-n, p-r-ui-s-i-sch
F-P	s-aint, e-rr-an-ce, r-a-b-a-tt-eu-r, b-u-r-eau-c-r-a-te
G-S	a-so-zi-a-le-re, e-be-ne, schnee-sturms, schnupft
E-S	bo-liv-i-a, id-i-ot, ring-side, scrunched
D-S	i-ni-ti-a-le, maan-zie-ke-re, kerst-staaf, traagst
G-M	er-barm-ung-s-los-ig-keit, titel-schrift, kinkerlitzchen
E-M	un-profess-ion-al-ly, im-patient-ly, un-do, quincentenary

Table 1. Examples of gold standard segmentations from different data sets. In the first column, G,E,F, and D stand for German, English, French, and Dutch, respectively. P, S, and M stand for phonology, syllabification, and morphology data, respectively.

maximum, and average *sizes* of the parts and the subsequent three columns to the minimum, maximum, and average string lengths. The last three columns give numbers relating to the size of the search space for full enumeration, which we determine via relationship (1). As concerns the size of the search space, i.e., the number of possible segmentations of strings x under these parameter values, we find that the number $S_A([\bar{n}], [\bar{k}])$, which gives the number of segmentations of the *average* string with an *average* number of parts, is usually quite small, ranging from 7 to 120 across the different data sets. Also, the 95 percent quantiles show that 95 percent of all strings, across the different datasets, admit at most a few hundred or a few thousand segmentations. The expected values are also moderate in size but the large standard deviations indicate that the distributions of the number of segmentations per string is very skewed, where a few strings allow very many segmentations. For example, the German noun *wahrscheinlichkeitsrechnung*, with length $n = 27$, admits $2,653,292$ segmentations with $k^* = 18$ parts, each between $\xi_{\min} = 1$ and $\xi_{\max} = 4$.

5. Experiments

For our experiments, we use as language model LM standard Ngram models, with modified Kneser-Ney smoothing, as implemented in the kylm language modeling toolkit.[3] We emphasize that we choose (discrete) Ngram models as language models merely for the sake of convenience and because Ngram models have a very strong tradition in NLP; other language models such as log-linear language models (Berger et al., 1996) or neural network language models (Bengio et al., 2001) might have been equally good (or better) alternatives. To contrast our methodology with the sequence labeling approach to sequence segmentation, we use conditional random fields (CRFs)

[3]Available at http://www.phontron.com/kylm/.

	k_{min}	k_{max}	$\bar{k}$	ξ_{min}	ξ_{max}	$\bar{\xi}$	n_{min}	n_{max}	$\bar{n}$
G-P-25K	1	27	8.66 ± 2.7	1	4	1.15 ± 0.4	1	31	9.97 ± 3.1
E-P-25K	1	20	7.18 ± 2.3	1	4	1.17 ± 0.4	1	22	8.37 ± 2.5
D-P-25K	1	25	9.09 ± 3.1	1	4	1.16 ± 0.4	1	29	10.53 ± 3.5
F-P-25K	1	18	6.69 ± 2.3	1	4	1.27 ± 0.5	1	20	8.50 ± 2.6
G-S-55K	1	10	3.62 ± 1.2	1	10	3.08 ± 1.1	1	31	11.15 ± 3.2
E-S-15K	1	7	2.43 ± 1.1	1	9	3.20 ± 1.3	1	19	7.80 ± 2.5
D-S-55K	1	11	3.51 ± 1.3	1	9	3.07 ± 1.0	1	30	10.78 ± 3.3
G-M-36K	1	9	2.40 ± 0.9	1	21	4.17 ± 2.1	1	31	10.01 ± 3.2
E-M-22K	1	5	1.68 ± 0.7	1	16	4.60 ± 2.1	1	21	7.73 ± 2.6

	$\lvert S_A([\bar{n}], [\bar{k}])\rvert$	$Q^{0.95}_{\lvert S_A(n,k)\rvert}$	$E[\lvert S_A(n,k)\rvert]$
G-P-25K	9	364	$465.52 \pm 28,058.0$
E-P-25K	7	105	27.55 ± 98.7
D-P-25K	45	364	$238.06 \pm 4,853.2$
F-P-25K	28	286	78.08 ± 504.8
G-S-55K	120	2,710	$1,547.04 \pm 56,553.2$
E-S-15K	7	210	59.99 ± 318.2
D-S-55K	120	2,430	$2,848.16 \pm 75,541.1$
G-M-36K	9	364	$365.17 \pm 31,063.1$
E-M-22K	7	45	10.43 ± 30.6

Table 2. Data sets (rows) and statistical properties. The set A *is* $\{\xi_{min}, \xi_{min}+1, \ldots, \xi_{max}\}$*. Description in text.*

(Lafferty et al., 2001) as a sequence labeling model SL, as implemented in the CRF++ toolkit.[4] Again, alternatives such as structured SVMs (Tsochantaridis et al., 2004) might have been equally well (or better) suited but we choose CRFs because of their reputation as yielding state-of-the art results on structured prediction problems in NLP. For all subsequent experiments, we use linear-chain conditional random fields with window size w (we include as features all character Ngrams that fit inside a window of $\pm w$ around the current character).

In our sequence labeling approach, we additionally consider another encoding scheme as the one indicated in Section 1. Namely, we also experiment on encoding the length of the segment directly in the labeling. For example, for the syllabic segmentation of *phoenix* as given in Section 1, this labeling would read as $0_1 0_2 0_3 0_4 1 0_1 0_2$ to represent the segmentation *phoen-ix*. Bartlett et al. (2008) have claimed that this *numbered encoding scheme* leads to better performance for the syllabification problem

[4] Available at `http://crfpp.googlecode.com/svn/trunk/doc/index.html`

because it biases the model to favor shorter segments. We refer to this labeling scheme as SL-NUM and to the (unnumbered) labeling scheme outlined in Section 1 as, simply, SL.

Generally, for all subsequent experiments, when indicating a dataset of size M, we perform ten-fold cross-validation to assess performance results, that is, our training data has size $0.9M$ for each of the ten folds. Throughout, as a performance measure, we use *word error rate*, the fraction of wrongly segmented sequences.

5.1. Phonological Segmentation

For phonological segmentation, we generate random samples of size $M = 25,000$ for German, English, Dutch, and French in the manner indicated in Section 4. We first assess, in Table 3, how well our SL modeling performs as a part prediction model PM. We see that k^*, the true number of parts of a given sequence, on average coincides with $\hat{k}$, the predicted number of parts, in about 97% of the cases for German, Dutch, and French, and in about 91% of the cases for English. Thus, if we used our LM models with $\hat{k}$, we would have error rates of at least 3% for German, Dutch, and French, and at least 9% for English. Higher upper bounds on performance can be reached by instead considering the intervals $B_1(\hat{k}) = \left\{\hat{k}-1, \hat{k}, \hat{k}+1\right\}$ wherein to search for k^*. In fact, as shown in the table, the probability that k^* is in $B_1(\hat{k})$ is considerably above 99% for all four datasets. These findings encourage us to use our *sequence labeling models SL as prediction models PM*.

	$P_{SL}[k^* = \hat{k}]$	$P_{SL}\left[k^* \in \left\{\hat{k}-1, \hat{k}, \hat{k}+1\right\}\right]$
German-25K	97.5 ± 0.25	99.8 ± 0.13
English-25K	90.9 ± 0.44	99.3 ± 0.27
Dutch-25K	96.5 ± 0.29	99.9 ± 0.08
French-25K	97.1 ± 0.26	99.9 ± 0.08

Table 3. Probability that k^* is identical to $\hat{k}$ as predicted by SL model or is in $B_1(\hat{k})$ in %. Phonology data.

Next, in Figure 1, we plot error rates in terms of N (for the LM Ngram models; we use $\hat{k}$ from the SL models). We see that for the LM-C models, performance levels off at about $N = 10$ or $N = 11$, while for the LM-W models, performance levels off already at $N = 6$ or $N = 7$. This is understandable as the word level models operate on entities of a larger size, namely, segments. We also usually see a convergence of both error rates as N gets larger. We omit similar graphs for window sizes w in the SL models,

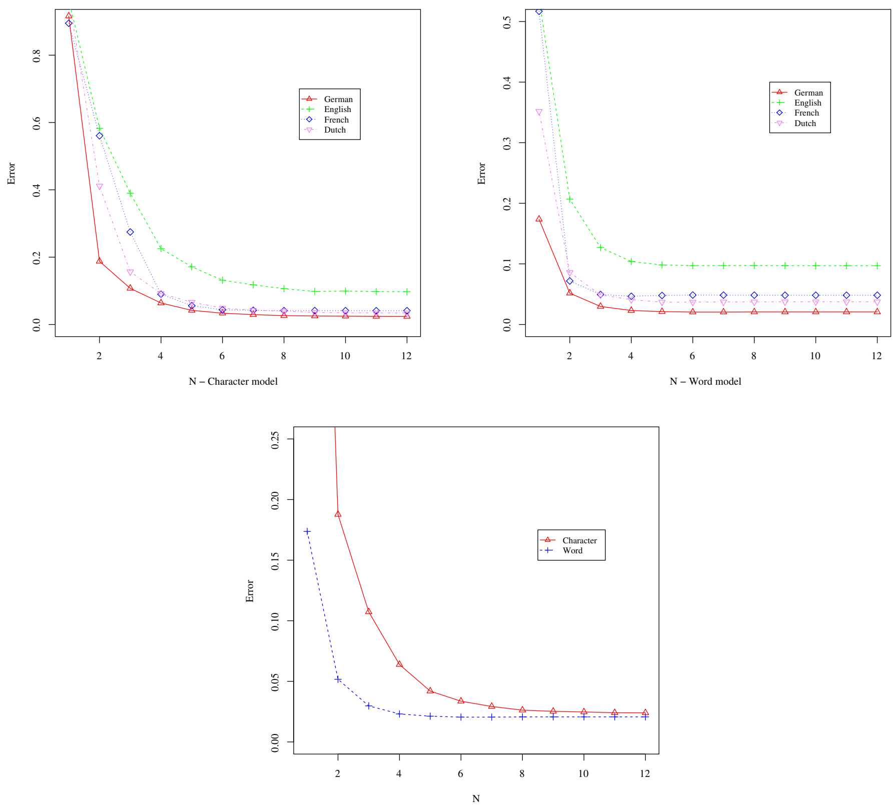

Figure 1. Performance of LM-C (top left) and LM-W (top right) as a function of N *in the Ngrams. Bottom: Character and word model in a single plot, exemplarily shown for German. Phonology data.*

but remark that a leveling off of performance occurs usually at $w = 4$ (note that this means that a context of $2w + 1 = 9$ characters is considered) or $w = 5$.

Now, based on these insights, we fix N at 10 for the LM-W models and at 11 for the LM-C models and use $w = 5$ for the SL models. We report results in Table 4. Throughout, we see that on our four datasets, the models SL-NUM and SL have no statistically significant different error rates, that is, on this data and for our CRF learning models, we cannot confirm that using the numbered coding scheme implies better performance results. Moreover, the two LM models have no statistically significant better performance than the SL models, too, when using as prediction for the number of parts the variables $\hat{k}$ from the SL models. In contrast, when enumerating all

	German-25K	French-25K	Dutch-25K	English-25K
SL-NUM$_{w=5}$	2.65 ± 0.27	3.64 ± 0.31	3.91 ± 0.32	10.10 ± 0.45
SL$_{w=5}$	2.68 ± 0.26	3.56 ± 0.27	3.86 ± 0.29	10.07 ± 0.45
LM-C$^{k=\hat{k}}_{N=11}$	2.73 ± 0.28	3.55 ± 0.27	3.87 ± 0.30	10.05 ± 0.49
LM-W$^{k=\hat{k}}_{N=10}$	2.74 ± 0.31	3.56 ± 0.25	3.88 ± 0.29	10.06 ± 0.40
LM-C$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\}}_{N=11}$	2.41 ± 0.32	4.09 ± 0.26	3.42 ± 0.32	9.78 ± 0.35
LM-C$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\},\beta=1.1}_{N=11}$	2.29 ± 0.27	3.39 ± 0.24	3.34 ± 0.31	9.15 ± 0.41
LM-W$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\}}_{N=10}$	2.06 ± 0.37	4.83 ± 0.33	3.74 ± 0.30	9.70 ± 0.48
LM-W$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\},\beta=1.1}_{N=10}$	2.09 ± 0.26	3.65 ± 0.31	3.44 ± 0.34	9.03 ± 0.49
LM-C$^{k=k^*}_{N=11}$	0.63 ± 0.25	1.22 ± 0.21	1.21 ± 0.09	3.01 ± 0.39
LM-W$^{k=k^*}_{N=10}$	0.60 ± 0.21	1.22 ± 0.19	1.23 ± 0.16	3.04 ± 0.35

Table 4. Error rates in % across different data sets and model specifications. Sub- and superscripts denote various parameterizations. Phonology data.

segmentations with number of parts k in $B_1(\hat{k})$ and selecting the highest scoring as predicted segmentation, performance results are, except for the French Brulex data, significantly better. For example, for the German, English and Dutch data, we find, for LM-C, error rate improvements of about 10%, 3%, and 11%, with regard to the SL models. Still larger improvements can be achieved by putting a *prior* on k. Note that, since the SL models are quite accurate PM models, it is more likely that $\hat{k}$ is correct than either $\hat{k}-1$ or $\hat{k}+1$. We experiment with a very simple heuristic that discounts the language model likelihood of segmentations with $\hat{k} \pm 1$ parts by a factor β. While selecting β by optimizing on a development set might lead to best results, we simply let $\beta = 1.1$ for all our data sets. This implies error rate improvements of about, for our best LM-C or LM-W models, 22%, 10%, 13%, and 5% for German, English, Dutch, and French, respectively, with respect to the SL models, where all improvements are statistically significant (paired two-tailed t-test, 5% significance level). Finally, as a kind of 'oracle model', we give performance results of our LM models under the assumption that the true number of parts k^* were known, for each given string to segment. We see, in this case, very large error rate improvements, of about 77%, 68%, 69%, and 65% for German, English, Dutch, and French, respectively, with respect to the SL models.

To say a word on the difference between the LM-C and LM-W models, we find that, a bit surprisingly, both models apparently perform, more or less, equally well (we would have expected the word level models to outdo the character level models, at least on the phonological segmentation task). In our case, the word level models perform better for German and slightly better for English, while this ordering is reversed for French and Dutch.

As concerns *running times*, on a 3.1 GHz processor, it takes around 18 min, over all 10 folds, for the CRFs to train, both for English (smallest string lengths, on average) and Dutch (largest string lengths, on average). Testing (decoding) takes about 2.39s for English and 3.69s for Dutch. In contrast, training the LM models takes around 42s for English and around 52s for Dutch. Generating all segmentations and evaluating them takes around 22s + 2min for English and 14min + 25min for Dutch when choosing $B_1(\hat{k})$ as search space. Thus, all in all, running times are quite moderate; also note that our segmentation and evaluation module are in Matlab (resp. Python) and Java, whereas the CRF is in C++. We also find that we search about 0.77% of the full search space (2^{n-1} segmentations per string of length n) and that if we explored the full search space, running times would be inflated by a factor of about 130 (hence, segmenting and evaluating 25,000 strings would take about 3 1/2 days for the Dutch data), with no (or almost no) increase in accuracy because $B_1(\hat{k})$ contains all (or almost all) correct segmentations (in fact, switching to, e.g., $B_2(\hat{k})$ implies no statistically distinguishable performance results, as we find).

We are not aware of any other study that would evaluate phonological sequence segmentation (but see also the related work section) and thus cannot compare our results here with those of others.

5.2. Syllabification

For syllabification, we use data set sizes as reported in Bartlett et al. (2008). In Table 5, we see that our SL model performs better here in predicting the correct number of parts of segmentations than in the phonological segmentation task, where the probability that the true k^* is in $B_1(\hat{k})$ is very close to 100% across the three languages.

	$P_{SL}[k^* = \hat{k}]$	$P_{SL}\left[k^* \in \left\{\hat{k}-1, \hat{k}, \hat{k}+1\right\}\right]$
German-55K	99.6 ± 0.09	100.0 ± 0.00
English-15K	96.7 ± 0.52	99.9 ± 0.03
Dutch-55K	99.4 ± 0.07	99.9 ± 0.01

Table 5. Probability that k^* is identical to $\hat{k}$ as predicted by SL model or is in $B_1(\hat{k})$ in %. Syllabification data.

While we omit an investigation of varying N in the Ngram models because of similarity of graphics with those previously shown, we mention that increasing N above

2 or 3 has no impact in the LM-W models since the average number of parts is much smaller here than in the phonological segmentation case (see Table 2); the same holds true for the morphological segmentation task below.

Thus, we fix N at 3 in the LM-W models and at 11, as before, in the LM-C models, giving results in Table 6. Again, we see performance increases of about, for German, English, and Dutch, respectively, 30%, 7%, and 42% for the best performing LM models over the SL models. Knowing the true k^* would, as before, yield still considerably better results. We report on an evaluation of the word level model only in the situation of a closed language model where the vocabulary stems from the training data (this excludes on the order of 5–10% of all test strings because some of their syllable parts never occurred in the training data, no matter the possible segmentation); in fact, the open vocabulary situation is uninformative since the LM-W model has huge error rates here, as our language model reserves so much probability mass for unseen words that segmentations with segments that do not occur in the training data are favored over those whose segment parts do occur there.[5] As we have expected, under the same conditions, the character level model still performs significantly better than the word level model in the case of syllabification. We omit an investigation of the numbered coding scheme, except for the English data, because of the huge increase in training time and since we find that this model actually never performs better than its unnumbered alternative.

Our results compare favorably with those reported by Bartlett et al. (2008), who claim to improve on competitors by a wide margin. Using an SL approach with a structured SVM as a labeling model, they obtain error rates of 1.19%, 10.55% (they give a result of 15.03% for SbA (Marchand et al., 2007)), and 1.80% for German, English, and Dutch, while we obtain 1.07%, 11.24%, and 1.49% here, with our best models. Thus, except for the English data, our results appear better, using the same training set sizes.[6] Concerning other results, Bartlett et al. (2008) cite error rates of Bouma (2003), using finite state techniques, of 3.50% for Dutch on 50K training instances, as we use, and 1.80% on 250K. For German, Demberg (2006)'s HMM approach achieves 2.13% on the *whole* of CELEX, which is double of our error rate. To our knowledge, our results are the best reported on the syllabification task for German and Dutch on the CELEX data and for our training set size of 50K.

5.3. Morphological Segmentation

Performance results for morphological segmentation are listed in Tables 7 and 8 (the Dutch data was unfortunately not available to us here). Again, our best perform-

[5]The same does *not* hold true for phonological segmentation, where parts are shorter and strings have more segments such that more reliable statistics can be computed.

[6] The better performance of Bartlett et al. (2008) on English may be due to the advantage of SVMs over standard Ngrams (and CRFs) at the small training set size for English; see He and Wang (2012) and our discussion below.

	German-55K	English-15K	Dutch-55K
SL-NUM$_{w=5}$	na	12.88 ± 0.70	na
SL$_{w=5}$	1.54 ± 0.21	12.14 ± 0.59	2.57 ± 0.16
LM-C$^{k=\hat{k}}_{N=11}$	1.20 ± 0.17	11.73 ± 0.69	1.63 ± 0.14
LM-W$^{k=\hat{k}}_{N=3}$	na	na	na
LM-C$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\}}_{N=11}$	1.41 ± 0.17	12.21 ± 0.72	1.77 ± 0.12
LM-C$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\},\beta=1.1}_{N=11}$	1.07 ± 0.15	11.24 ± 0.62	1.49 ± 0.06
LM-C$^{k=k^*}_{N=11}$	0.82 ± 0.11	9.40 ± 0.58	1.14 ± 0.08
LM-C$^{\text{vocab},k=\hat{k}}_{N=11}$	1.49 ± 0.13	14.95 ± 0.92	1.71 ± 0.16
LM-W$^{\text{vocab},k=\hat{k}}_{N=3}$	3.53 ± 0.22	18.82 ± 1.59	3.49 ± 0.23

Table 6. Error rates in % across different data sets and model specifications. Sub- and superscripts denote various parametrizations. Syllabification data.

	German-36K	English-22K
SL-NUM$_{w=5}$	na	13.45 ± 0.31
SL$_{w=5}$	16.34 ± 0.43	11.68 ± 0.50
LM-C$^{k=\hat{k}}_{N=11}$	15.15 ± 0.60	11.18 ± 0.47
LM-W$^{k=\hat{k}}_{N=3}$	na	na
LM-C$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\}}_{N=11}$	11.08 ± 0.49	9.50 ± 0.72
LM-C$^{k\in\{\hat{k}-1,\hat{k},\hat{k}+1\},\beta=1.1}_{N=11}$	11.86 ± 0.60	9.31 ± 0.69
LM-C$^{k=k^*}_{N=11}$	2.31 ± 0.34	0.98 ± 0.13
LM-C$^{\text{vocab},k=\hat{k}}_{N=11}$	6.85 ± 0.68	3.60 ± 0.38
LM-W$^{\text{vocab},k=\hat{k}}_{N=3}$	6.88 ± 0.70	3.62 ± 0.40

Table 7. Error rates in % across different data sets and model specifications. Sub- and superscripts denote various parametrizations. Morphology data.

ing LM models are about 32% and 20% better, for German and English, respectively, than the SL approach. Concerning error rates, we omit a comparison with other work because most approaches in morphological segmentation are unsupervised, and we in fact are not aware of supervised performance results for the data we consider.

	$P_{SL}[k^* = \hat{k}]$	$P_{SL}\left[k^* \in \left\{\hat{k}-1, \hat{k}, \hat{k}+1\right\}\right]$
German-36K	85.6 ± 0.44	98.9 ± 0.17
English-22K	89.1 ± 0.49	99.7 ± 0.05

Table 8. Probability that k^* is identical to $\hat{k}$ as predicted by SL model or is in $B_1(\hat{k})$ in %. Morphology data.

5.4. Discussion

To say a word on exhaustive enumeration as a solution technique to optimization problems, beginner's courses to combinatorial optimization usually emphasize that exhaustive search is the *simplest* and *most straightforward* approach to any optimization problem that admits only finitely many possible solution candidates; and that it is, if feasible at all (i.e., from a complexity perspective), also guaranteed to lead to *optimal solutions* (Nievergelt, 2000). Hence, if the segmentation problem in NLP was framed as the problem of finding the segmentation s_n of sequence $x = x_1 \dots x_n$ that solves

$$\arg\max_{s_n \in \mathcal{S}(n)} f_\theta(s_n),$$

i.e., for *model* f_θ *and model parameter vector* θ *given* (if one wants so, this is the *decoding problem* for sequence segmentation), then our approach to sequence segmentation would surely be optimal, provided that our search space restrictions are not critical. The problem in natural language processing (NLP) is, of course, that we neither know the most appropriate (or 'true') model f_θ for our task nor, in statistical NLP, do we know the true parameter vector θ. The scope of this work is neither model selection nor feature engineering (determination of a good model f_θ), however, nor is it the estimation of the parameter vector θ. What we intend to show, instead, is that, for our problem tasks, efficient enumeration is generally feasible such that, *for* f_θ *given*, our approach is optimal. Thus, to summarize, if a technique performed better than the approach sketched in this work, it must be due to a superior model f_θ (e.g., than our standard Ngrams),[7] and not due to *search*, as we focus on. Here, we content ourselves, however, with the fact that standard Ngrams in conjunction with (almost) exact search can, as shown, outperform state-of-the-art approaches to sequence segmentation (this includes, at least on two out of the three data sets on the syllabification task, structured SVMs, which appear to be the *primus inter pares* among current

[7]Or due to 'better' estimation of θ.

sequence labeling methods; see the discussion below), rendering the investigation of better models f_θ momentarily superfluous.

To contrast our approach with other methods, many sequence labeling algorithms, for example, rely on crucial *restrictions with regard to allowable scoring functions* f_θ, as mentioned. For example, most graphical models assume Markov-type independence assumptions for the label sequences. In contrast, with our approach, f_θ may be arbitrary, and arbitrarily complex. To make this feasible, we instead *restrict search space*, as outlined. Moreover, as Tables 3, 5, and 8 demonstrate, the search space we prune away has very little probability of actually containing the correct segmentations (we could easily lower this probability to zero by, e.g., considering the search spaces $B_2(\hat{k})$) such that our restrictions may not affect accuracy at all, while pruning model complexity may be more expected to yield sub-optimal performance. Our approach may also be seen in the context of *coarse-to-fine decoding* procedures: first, we use a sub-optimal model f_θ^1 to restrict search space, and then use any arbitrary, 'superior' models f_θ^2 in conjunction with full enumeration on the restricted search space to improve on f_θ^1; we have shown how and that such a procedure can be made effective within the field of sequence segmentation for selected NLP applications.

We also note that for specific f_θ, e.g., when f_θ is *decomposable* (Terzi, 2006), full enumeration may not be necessary because efficient dynamic programming (DP) solutions apply. For example, for word level Ngrams, a simple DP solution whose running time is quadratic in n, sequence length, can be given when $N = 1$. In contrast, our approach works for *any* f_θ, not only for decomposable models.

6. Related Work

Phonological segmentation may be a crucial step in, e.g., grapheme-to-phoneme conversion (G2P) models based on many-to-many alignment approaches (Jiampojamarn et al., 2007, 2008; Bisani and Ney, 2008; Eger, 2012), where, for decoding, grapheme strings need to be segmented. Jiampojamarn et al. (2007) employ instance-based learning for this 'letter chunking task', without, however, evaluating their model's performance (they solely evaluate G2P performance); the same holds true for the three other papers cited. Of course, sequence segmentation similar to phonological segmentation may play a key role in string transduction problems, including lemmatization, stemming, etc., in general (Dreyer et al., 2008). As concerns syllabification, besides 'rule-based approaches' (see the discussion and references in Marchand et al., 2007), in the statistical context, we are aware of Bartlett et al. (2008)'s sequence labeling approach and a lazy learning segmentation-by-analogy framework due to Marchand et al. (2007); older approaches include neural network backpropagation learning (Daelemans and van den Bosch, 1992) or finite-state techniques (Bouma, 2003). Intriguingly, syllabification may prove beneficial for solving the G2P task, as Bartlett et al. (2008) demonstrate; its most obvious application is, of course, to provide candidates for hyphenation. There is a huge literature on morphological segmentation, e.g., Creutz and Lagus (2007); Poon et al. (2009), but most approaches are unsupervised here. As concerns applications of morphological segmentation, besides serving for quantitative analyses such as morpheme counts in texts, it may serve as a preprocessing step for phonological segmentation and/or syllabification.

The literature on CRFs, as we have used as a SL model, is vastly expanding, too; among the most interesting developments in our context are probably semi-Markov CRFs (Sarawagi and Cohen, 2004), which explicitly segment the input sequence. An analysis within our context would be scope for future research. Stoyanov and Eisner (2012) discuss approximate inference and decoding for higher-treewidth graphical models underlying CRFs. A recent comparison of state-of-the-art sequence labeling approaches is given in He and Wang (2012) where it is shown that structured SVMs outperform competitors on tagging and OCR; performance differences decrease, however, in data set size.

7. Concluding Remarks

Our contribution to the *mathematics* of linguistics is to relate the sequence segmentation problem to restricted integer compositions, which have attracted increasing interest in mathematical combinatorics recently — not the least because of their relationship to *extended binomial coefficients*. Our contribution to *computational* linguistics is to show that exhaustive enumeration of sequence segmentations is, for an array of interesting segmentation problems in NLP, cheap, given adequate restriction of search space, such that exact search for the optimal segmentations can easily be conducted, for arbitrary evaluation models f_θ. We also show that for the simple choice of f_θ as standard Ngram models, performance results on par or better than current state-of-the-art sequence labeling approaches can be achieved.

In future work, different language models f_θ, possibly including *global features*, are worthwhile investigating, among other things, as well as interpolating of character and word level language models.

Bibliography

Baayen, R. Harald, Richard Piepenbrock, and Leon Gulikers. The CELEX2 lexical database, 1996.

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

Bender, Edward A. and E. Rodney Canfield. Locally restricted compositions, I. Restricted adjacent differences. *The Electronic Journal of Combinatorics*, 12, 2005.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. In *NIPS 13*, pages 933–938, 2001.

Berger, Adam L., Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, Mar. 1996. ISSN 0891-2017.

Bisani, Maximilian and Hermann Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Commun.*, 50(5):434–451, May 2008. ISSN 0167-6393. doi: 10.1016/j.specom.2008.01.002.

Bouma, Gosse. Finite state methods for hyphenation. *Nat. Lang. Eng.*, 9(1):5–20, Mar. 2003. ISSN 1351-3249. doi: 10.1017/S1351324903003073.

Content, Alain, Philippe Mousty, and Monique Radeau. Brulex. Une base de données lexicales informatisée pour le français écrit et parlé. *L'année psychologique*, 90(4):551–566, 1990. ISSN 0003-5033. doi: 10.3406/psy.1990.29428.

Creutz, Mathias and Krista Lagus. Unsupervised models for morpheme segmentation and morphology learning. *ACM Trans. Speech Lang. Process.*, 4(1):3:1–3:34, Feb. 2007. ISSN 1550-4875. doi: 10.1145/1187415.1187418.

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

Demberg, Vera. Letter-to-phoneme conversion for a german text-to-speech system, 2006.

Dreyer, Markus, Jason R. Smith, and Jason Eisner. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, EMNLP '08, pages 1080–1089, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1613715.1613856`.

Eger, Steffen. S-restricted monotone alignments: Algorithm, search space, and applications. In *Proceedings of Coling*, 2012.

Eger, Steffen. Restricted weighted integer compositions and extended binomial coefficients. *Journal of Integer Sequences*, 2013.

Fahssi, Nour-Eddine. A systematic study of polynomial triangles. *The Electronic Journal of Combinatorics*, 2012.

Goldwater, Sharon, Thomas L. Griffiths, and Mark Johnson. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54, July 2009. ISSN 00100277. doi: 10.1016/j.cognition.2009.03.008.

He, Zhengyan and Houfeng Wang. A comparison and improvement of online learning algorithms for sequence labeling. In *Proceedings of Coling*, 2012.

Heubach, Silvia and Toufik Mansour. Compositions of n with parts in a set. *Congressus Numerantium*, 164:127–143, 2004.

Jiampojamarn, Sittichai, Grzegorz Kondrak, and Tarek Sherif. Applying many-to-many alignments and Hidden Markov Models to letter-to-phoneme conversion. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007)*, pages 372–379, Rochester, New York, Apr. 2007. Association for Computational Linguistics.

Jiampojamarn, Sittichai, Colin Cherry, and Grzegorz Kondrak. Joint processing and discriminative training for letter-to-phoneme conversion. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, pages 905–913, June 2008.

Lafferty, John D., Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1.

Malandro, Martin E. Integer compositions with part sizes not exceeding k, 2011. Preprint available at `http://arxiv.org/pdf/1108.0337.pdf`.

Marchand, Yannick, Connie Adsett, and Robert Damper. Evaluation of automatic syllabification algorithms for English. In *Proceedings of the 6th international speech communication association (ISCA)*, 2007.

Nievergelt, Jürg. Exhaustive search, combinatorial optimization and enumeration: Exploring the potential of raw computing power. In *Proc. Conf. on Current Trends in Theory and Practice of Informatics*, pages 18–35, 2000.

Opdyke, John Douglas. A unified approach to algorithms generating unrestricted and restricted integer compositions and integer partitions. *Journal of Mathematical Modelling and Algorithms*, 9(1):53–97, 2010.

Page, Daniel R. Generalized algorithm for restricted weak composition generation. *Journal of Mathematical Modelling and Algorithms (JMMA)*, pages 1–28, 2012. ISSN 1570-1166. doi: 10.1007/s10852-012-9194-4. Published online July 20.

Poon, Hoifung, Colin Cherry, and Kristina Toutanova. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-2009, pages 209–217, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-41-1.

Sarawagi, Sunita and William W. Cohen. Semi-Markov conditional random fields for information extraction. In *Proceedings of NIPS*, 2004.

Shapcott, Caroline. $\mathcal{C}$-color compositions and palindromes. *Fibonacci Quarterly*, 50:297–303, 2012.

Stoyanov, Veselin and Jason Eisner. Minimum-risk training of approximate CRF-based NLP systems. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 120–130, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. ISBN 978-1-937284-20-6.

Terzi, Evimaria. *Problems and Algorithms for Sequence Segmentation*. PhD thesis, University of Helsinki, 2006.

Tsochantaridis, Ioannis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st international conference on Machine Learning (ICML)*, pages 823–830, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5. doi: 10.1145/1015330.1015341.

Van den Bosch, Antal, Stanley Chen, Walter Daelemans, Bob Damper, Kjell Gustafson, Yannick Marchand, and Francois Yvon. Pascal letter-to-phoneme conversion challenge, 2006. URL `http://www.pascalnetwork.org/Challenges/PRONALSYL`.

Address for correspondence:
Steffen Eger
`eger.steffen@gmail.com`
Goethe University
Grüneburgplatz 1
60323 Frankfurt am Main, Germany