30 years of the Dutch Language Union

Linde van den Bosch

Nederlandse Taalunie

It is a great honour for me to address an audience of such renowned computational linguists. I would like to thank the organising committee of CLIN 2010 for this opportunity. The CLIN conference is held every year at a university in either the Netherlands or Belgium. It provides researchers in the field of computational linguistics with a unique opportunity to present and discuss their research. Belgium and the Netherlands are neighbours with much more in common than just a border. What is particularly important in relation to this conference is that the two countries share a common language: Dutch.

It is this bond which, in 1980, led to the two countries joining forces to set up the Dutch Language Union, with the aim of implementing common language policy for Dutch. In 2004, the two founding members were joined by Surinam as an associate member of the Dutch Language Union (DLU).

DLU initiatives cover all aspects of language policy. Each one is aimed at creating the right conditions to make it easier for Dutch speakers to use their language in as many different situations as possible, including when they are abroad.

It is ultimately not the governments of the Netherlands, Flanders and Surinam, who are the DLU's most important 'clients', but the people who use Dutch to communicate.

The activities of the DLU range from promoting the teaching of Dutch, both within the Dutch speaking areas and beyond, and stimulating language-related cultural and literary cooperation, to compiling dictionaries and grammars. Over the last decade the DLU has taken a serious interest in supporting the development of digital language resources and human language technologies (HLT) for Dutch.

As Dutch is a so-called medium-sized language and companies are not always willing or able to invest in developing HLT for a language with a relatively small market, government support was needed. On the other hand, the development of HLT is considered essential, if a language is to survive in the information society.

It was against this background that the DLU set up a number of initiatives aimed at strengthening the position of Dutch in human language technologies. The most important being:

- The STEVIN programme: a subsidised programme aimed at establishing a complete digital language infrastructure for Dutch, and promoting strategic research in language and speech technology

- The HLT Agency: a central repository for digital language resources based

Proceedings of the 20th Meeting of Computational Linguistics in the Netherlands
Edited by: Eline Westerhout, Thomas Markus, and Paola Monachesi.

within an important linguistic organisation in our language area: the Institute for Dutch Lexicology (INL).

I am delighted that a number of STEVIN results are being presented at this year's CLIN conference.

As the Dutch Language Union is a relatively small organisation, these activities have been set up in close co-operation with the relevant ministries and organisations in Belgium and the Netherlands. This is a key characteristic of the way we work: the DLU does not operate in isolation, but in close cooperation with other professional organisations and associations both within and outside of the Dutch language area. It is thanks to such cooperation that we are today able to look back on the past 30 years, and conclude that a great deal has been achieved, since the DLU was founded in 1980:

- In total there are more than 400,000 learners of Dutch as a foreign language across the world, and Dutch is taught at 200 universities in 40 different countries
- Books by Dutch and Flemish authors are translated into 100 languages
- Close cooperation links have been established with Surinam, the Dutch Antilles, Aruba, South Africa and Indonesia
- Advice on a range a Dutch language and linguistic issues is freely available to the public. In 2009, 5.5 million items were consulted, and 6,300 new questions were submitted and answered
- The DLU website, "Taalunieversum", receives over 17 million visitors a year
- Numerous digital Dutch language resources have been made available to researchers and to the general public. These resources are now being managed and maintained for future use.

2010 marks the 30th anniversary of the DLU, and also the 20th anniversary of the CLIN conference. I would like to congratulate the CLIN community for having reached this milestone. We should acknowledge the valuable contribution CLIN has made towards creating an excellent environment in which Dutch and Belgian computational linguists can exchange and cultivate ideas. It is in part thanks to CLIN that computational linguists, like yourselves, have come to enjoy such international renown.

I would like to conclude with one simple wish: in 10 years' time, when the DLU celebrates its 40th anniversary and CLIN its 30th, my speech will not need to be translated beforehand, but will be able to be produced real-time, using machine translation.

I hope you all enjoy an interesting and fruitful conference.

Improving Successor Variety for Morphological Segmentation

Çağrı Çöltekin

University of Groningen

Abstract

Successor variety is a commonly used measure for segmentation in language processing. It is based on a simple idea that a large variety of letters (or phonemes) following an initial word (or utterance) segment indicates a possible boundary. It dates back to Harris (1955), and several methods based on successor variety have been used in the literature, particularly for the purpose of segmenting words into morphemes. However, there have not been many studies analyzing the measure itself. Even though the idea is simple and effective, the current use in the literature does not utilize the measure to its full extent due to a number of problems with the successor variety scores. This paper intends to address these problems by introducing a normalization method, and demonstrates—using segmentation experiments on two typologically different languages— the effectiveness of this improvement on the morphological segmentation task.

1 Introduction

Segmentation is a prominent problem in language processing. Spoken language input does not contain reliable word boundary markers like white spaces in most writing systems. Even the writing systems that utilize word boundary markers do not mark all linguistically relevant boundaries, such as morpheme boundaries. Humans, as well as computers, need to segment the continuous input into linguistic units such as words and morphemes to be able to interpret the input. This task becomes more important, and more difficult, where the system in question tries to learn these units. In the segmentation task, the competent language users (adult humans or computational systems with linguistic knowledge built in) are aided by rich linguistic information. Although the segmentation task is still difficult, existing linguistic generalizations, e.g., a comprehensive lexicon, are useful for segmentation. However, infants acquiring language or the data driven computational systems do not have that luxury. To be able to acquire a lexicon, learners have to first deal with the segmentation task without a lexicon. This paper analyzes a commonly used measure for segmentation, *successor variety* (SV), first proposed by Harris (1955). We introduce the SV in the context of segmenting written words to morphemes and suggest a simple but effective improvement. To demonstrate the efficiency of our proposed solution, we present a number of experiments on two languages with differing typology.

The successor variety is not a clearly defined algorithm. Several different segmentation algorithms based on the SV have been used with varying success in the literature. Except Hafer and Weiss (1974) and Bordag (2005) there have not been

Proceedings of the 20th Meeting of Computational Linguistics in the Netherlands
Edited by: Eline Westerhout, Thomas Markus, and Paola Monachesi.

many attempts to analyze and improve the method. This paper first presents an in-depth analysis of the method, then proposes a simple and effective improvement to get more out of it.

Harris (1955) initially proposed the successor variety for segmenting transcribed spoken language utterances into morphemes. The idea is simple: morpheme boundaries are suggested after the utterance segments that may be followed by a large variety of phonemes. In more recent work, however, the measure found its use particularly in segmenting words into morphemes (Hafer and Weiss 1974, Déjean 1998, Al-Shalabi et al. 2005, Bordag 2005, Goldsmith 2006, Bordag 2007, Demberg 2007, Stein and Potthast 2008). Since these studies focused on written text, the measure is frequently referred to as *letter successor variety* (LSV). However, it is equally applicable to other basic (linguistic) units such as phonemes.

The SV based methods are closely related to a number of other approaches based on *entropy*, *(un)predictability*, *surprisal* and *mutual information*. The basic rationale behind these methods is that in a continuous stream formed by concatenating a number of repeating units, such as words or morphemes, *predictability of next (or previous) letter is higher (low entropy) within the units, lower (high entropy) between the units*. Besides linguistic data, this happens to be valid for a wide variety of naturally occurring streams (Cohen et al. 2007). Similar ideas have also been exploited by computational models of human language acquisition. *Simple recurrent networks* used for learning a large variety of linguistic phenomena are trained to guess the next unit in the input sequence, and the networks function based on how predictable the next unit in the test input is. Brent (1996) used *distributional regularities* where a segmentation decision is made at points with unexpected sequences of phonemes. And, indeed, children have been shown to be sensitive to this type of information in the continuous speech stream very early in life (Saffran et al. 1996).

The subject of this paper, the SV method, does not utilize all possible information that can be useful for segmentation task. There are several other sources of information, or cues, that can facilitate the task of learning segmentation. When available, existing *lexical knowledge*; the *words encountered in isolation*; *prosodic cues*, such as word stress; *phonotactic constraints*, such as the phoneme sequences that do not occur beginning or end of the words; *allophonic differences*; and *vowel harmony* are known to be useful for segmentation. However, some of these are not always available, some require a working knowledge of lexical items of the language to be useful. On the other hand, the SV (and similar methods) uses the relationship between the successive basic units, and it is applicable as long as basic units can be discriminated.

We refer to the methods listed so far as *local* methods. In local methods, the segmentation decision is made only by checking the immediate context. Local methods contrast with a large number of successful segmentation algorithms that are based on optimizing a *global* objective function. Global optimization based methods typically rely on two competing factors: one preferring the smaller units, hence segmentation; the other not allowing over-segmentation. The ‘best’ segmentation is, then, the one found by optimizing the combination of these factors. Examples of these include models based on the minimum description length (MDL) (Goldsmith 2001, Goldsmith 2006), maximum a posteriori (MAP) estimate (Creutz and Lagus 2007), or explicit probabilistic (Bayesian) models (Brent 1999, Goldwater et al. 2009).

Most recent approaches to segmentation make use of a number of available cues, and sometimes, a combination of local and global methods. This paper focuses on only one of the local methods, the SV, and suggests an improvement to this method as well as provides an in-depth empirical analysis. The next section will introduce the method. After reviewing related studies in Section 3, we will demonstrate the use of the method on real world data in Section 4. Section 5 describes the suggested improvement. We present a number of experiments demonstrating the effectiveness of the improvement on large word-lists for English and Turkish in Section 6. Section 7 concludes after a brief discussion of the results and other possible improvements.

2 Successor Variety

The successor variety of a string is the number of distinct phonemes (or letters) that can follow the string in the language we are interested in. Harris' use of successor variety was aimed at finding morphemes in spoken language. Given an initial segment of an utterance, a high number of phonemes that may follow the segment was used as an indication of a morpheme boundary. His definition of 'high' number of phonemes depended on human judgments.

We define successor variety as the number of distinct letters[1] that follow a given initial word segment as counted in a word list. More formally, for a suffix[2] x, a letter y and a word list W formed by letters in alphabet A,

$$SV(x) = \sum_{y \in A} c(x, y)$$

where,

$$c(x,y) = \begin{cases} 1 & \text{if string } xy \text{ occurs as an initial word segment in W} \\ 0 & \text{otherwise} \end{cases}$$

For example, given the word list in Figure 1a, and the test word `reading`, Figure 1b presents successor values for the test word. Note that we assumed a hypothetical end-of-word letter after the word `read`. For this small list, the successor values (SV) after `rea-` and `read-` are higher (2 and 3 respectively), potentially indicating morpheme boundaries.

[1] The method and the improvement we suggest are applicable to other units. Our definition is based on letters, as we will only use written language data in this study.

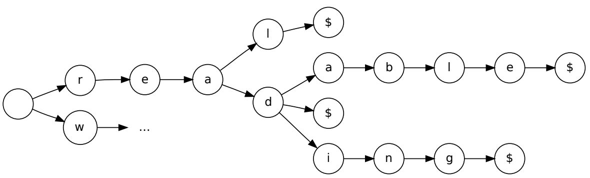
[2] Our use of the terms 'prefix' and 'suffix' in this article refers to any initial or final word segment. These segments are not necessarily linguistic units, i.e., morphemes attached to the beginning or the end of the words. We will clearly state it when these terms are used to refer to linguistic units.

read	readable	reading	real
write	writing	writer	working

(a)

letter		**r**		**e**		**a**		**d**		**i**		**n**		**g**	
SV	2		1		1		2		3		1		1		1
PV	1		1		1		1		3		1		1		5

(b)



(c)

Figure 1: (a) An example word list. (b) the successor and predecessor values for the test word `reading`. (c) A part of the trie structure representing the given word list. The character '$' represents the hypothetical end-of-word character.

A straightforward extension of the SV is the *predecessor value* (PV), the number of different letters that can precede a certain suffix. The line labeled 'PV' in Figure 1b lists the predecessor values for our example.

The successor and predecessor values for a given word list can easily be computed making use of a data structure called *prefix tree*, or *trie*. If the input word list is inserted into a trie, the successor value for a given prefix is the number of branches after the corresponding node. Figure 1c presents part of the trie for our example.

Intuitively, the successor and predecessor values seem to be simple and usable indications of morpheme boundaries. However, how can we use them to actually find the morpheme boundaries? How can we avoid finding non-morphemes like `rea` in our test word list, and not miss the real morpheme `write`? Given our example word list, `write` has the same SV and PV as the non-morpheme `rea`. What changes with the size of the word list, or language? After reviewing how others dealt with these questions, the rest of the paper tries to fill some of the gaps in answers to these questions.

3 Related Work

Most of the studies in the literature are 'consumers' of the SV based methods (e.g., Déjean 1998, Al-Shalabi et al. 2005, Goldsmith 2006, Demberg 2007, Stein and

Potthast 2008)). All these studies use the measure in one way or another to find morpheme boundaries in words. To our knowledge, there are only two studies that try to analyze and improve the method. The most elaborate study of various SV options is by Hafer and Weiss (1974), and the work by Bordag (2007) combines the SV with other methods to increase its effectiveness.

The goal of Hafer and Weiss (1974) (H&W) is stemming for information retrieval purposes. However, a good part of the paper investigates various options to segment words into morphemes including some variations of the SV method. H&W ran 15 different experiments on English and reported results from 13 of them. The criteria to segment in their experiments depended on a combination of SV and/or PV together with corresponding thresholds (or cutoff values); SV or PV being on a peak or plateau; the suffix or prefix at a certain location being a word by itself; and analogous to SV and PV, *successor entropy* (SE) and *predecessor entropy* (PE). In this paper, we will only focus on the variety scores. However, we will give a brief description of the entropy method, since the entropy scores may be more suitable for certain applications (where precision is more favorable to recall), and the normalization approach we advocate in this paper is also applicable to entropy values.

The segmentation entropy of a given prefix x is defined as:

$$H(x) = -\sum_{y \in succ(x)} \frac{f(xy)}{f(x)}\, log_2 \frac{f(xy)}{f(x)}$$

where, $f()$ returns the frequency (count) of the words starting with the given prefix, xy is the string that is formed by concatenation of string x and the letter y, and $succ()$ returns all successor letters for the given prefix. This formula gives the SE. One can easily obtain PE by modifying $successor$ with $predecessor$ and changing the order of the concatenation.

As H&W also mention, compared with the SV values, the entropy values provide an averaging effect that may reduce noise. However, it also reduces the sensitivity to the real boundaries. Hence, one expects better accuracy from entropy values, while the SV values provide better coverage. To clarify the difference between the entropy and the SV values, consider two hypothetical cases: a particular prefix is followed by (i) 2 different suffixes each starting with different letters, (ii) 5 different suffixes where 4 of them share the same first letter. For both cases SV will be 2, however, SE will be higher for case (i) compared to case (ii).

The best performing methods chosen by H&W use a complex combination of threshold values for variety or entropy scores with the additional knowledge on whether parts of the word occur in the word list as a standalone word or not. Two best performing methods selected by H&W are given below:

- $(\mathrm{PW}\ \textit{and}\ \mathrm{PV}_i > 5)\ \textit{or}\ (\mathrm{PV}_i > 17\ \textit{and}\ \mathrm{SV}_i > 2)$
- $((\mathrm{SE}_{i-1} = 0)\ \textit{and}\ (\mathrm{SE}_i > 0.8\ \textit{or}\ \mathrm{PE}_i > 0.8))\ \textit{or}$
  $((\mathrm{SE}_{i-1} \neq 0)\ \textit{and}\ ((\mathrm{PW}\ \textit{and}\ \mathrm{PE}_i > 0.8)\ \textit{or}\ (\mathrm{PE}_i > 3.0\ \textit{and}\ \mathrm{SE}_i > 1.0)))$.

where, ‘PW’ stands for ‘prefix is a standalone word’ in the word list.

letter		**r**		**e**		**a**		**d**		**i**		**n**		**g**	
SV	27		16		27		10		10		2		2		4
PV	7		5		16		10		25		5		13		28

Figure 2: The SV and PV values for the word `reading` calculated from the CELEX database.

The criteria are complicated. However, they seem to work well, and surprisingly, Al-Shalabi et al. (2005) report that the same criteria work also well for Arabic, a language with different morphological properties.

The most important problems with these criteria are their complexity and the high number of tunable parameters. Even though they seem to also work in different languages (this will be discussed further in Section 4), it is likely that the parameters are language (or even corpus) dependent.

Bordag (2007) uses a different method to improve the SV based measures. Finding morpheme boundaries in the example we have presented in Figure 1 is relatively easy due to the choice of words. As we will present later, finding boundaries on the SV values calculated on a large collection of unrelated words is more difficult. Realizing that, Bordag first does a contextual similarity analysis, and finds a relatively small number (e.g., 150) of words with high probability of semantic relatedness to the target word. The SV values are calculated on this smaller set of words, and similar to H&W, segmentation decision is based on thresholds.

Besides the thresholds for SV or PV values, Bordag's method introduces a new parameter, the number of related words. Additionally, the method's success depends highly on the contextual similarity analysis, and the corpora used for the similarity analysis.

In this study, we introduce a new approach to improve the SV based methods. We propose a method to normalize the SV scores that increases the effectiveness of the method, and allows for an easier interpretation of the scores. However, before introducing the normalization procedure, we will have a closer look at the SV values calculated on real-world data.

4 Successor Variety in Real World Data: a Closer Look

So far, the examples we used were toy examples demonstrating the method. However, the language data in the real world do not come in neatly organized small packages of word lists. In this section we will try to demonstrate some of the general characteristics of the SV measures as seen in real-world examples.

The experiments reported in this paper have been done on two typologically different languages: English and Turkish. For the English data we used the CELEX database (Baayen et al. 1995). As CELEX does not provide standard surface form segmentations that we need for evaluating our methods, we used the segmentations provided by Hutmegs (Creutz and Lindén 2004). The Turkish word list is extracted from the METU Corpus (Say et al. 2002). The Turkish gold-standard is obtained with the help of a finite-state morphological analyzer (Çöltekin 2010). The number of word types in the English data set is 114184 and the number of words in the Turkish data set is 143275. For the experiments, where using the same number of words is important, we randomly removed 29091 words with frequency one from the Turkish word list.

To get a first idea, we repeat the Figure 1b in Figure 2 this time calculating the values from the complete CELEX word-form list. Predecessor value peaks after `-ing` (traversing right to left). However, we do not see the same on successor values. The only peak value we identify with successor values is after `re-` which is not desirable for this word, but `re-` being a common prefix in English, this is an expected result. If we were to segment successor and predecessor peak values, we would (mistakenly) segment the word as `re-ading`.

The SV and PV values in Figure 2 also demonstrate a general trend: the SV values tend to be higher at the beginning and PV values tend to be higher at the end of the words, and they decrease as they go right and left respectively. To visualize this trend we plotted the mean SV and PV values in Figure 3. In addition to the average SV and PV for the CELEX and METU word lists, Figure 3 also presents a random baseline that we will explain in detail in Section 5. Clearly, the SV values are high at the beginning and the PV values are high at the end of the word (note that in the PV graph the index values are reversed). Except for the separation on the x-axis due to the average word length, and the height of the graphs (partially due to the size of the alphabet), the graphs are rather similar.[3]

Figure 3 indicates a clear problem with the strategies using threshold values and peaks for segmentation. Due to the exponential drop of the SV (and PV), it is difficult to find threshold values that would work everywhere. As the SV values are naturally high at the beginning of the words, a small threshold will suggest incorrect boundaries at the beginning of the words. If the threshold is tuned not to make mistakes at the beginning of the words, then the threshold value will be too conservative for the rest of the word.

The problem affects the performance at the beginning (for SV) and end (for PV) of the words. Unfortunately, most suffixes and prefixes in natural languages are rather short and attach to the end or beginning of the words. Hence, not accounting for these tendencies will make SVs only useful for detecting suffixes, and PVs for prefixes, where otherwise combination of both values may yield a better classification. Luckily, the problem can at least partially be fixed by collecting some simple statistics over the data. The next section will present the proposed solution for this problem and the effectiveness of this solution.

Figure 3 also reveals a difference between the distribution of the SV and PV

[3] Both graphs for Turkish show a clear raise after position 2, causing a peak at 3. This is due to a phenomenon we briefly discuss in Section 7. Turkish has 21 consonants, and 8 vowels. In the Turkish word list, 80% of the first letters are consonants. The probability of seeing a vowel after a consonant is 0.84. Hence, it is more likely to see a vowel as the second letter. Since there are only 8 vowels, as with after any consonant, the SV values after the first letter tend to be low. Similarly, the SV values after the second letter, as with after any vowel, tend to be high. The distribution of the letters at the end of the words, hence the PV values, also follow a similar pattern. For English, having more consonant-consonant and vowel-vowel bigrams, the same effect is observed in the graphs only as a slope change.

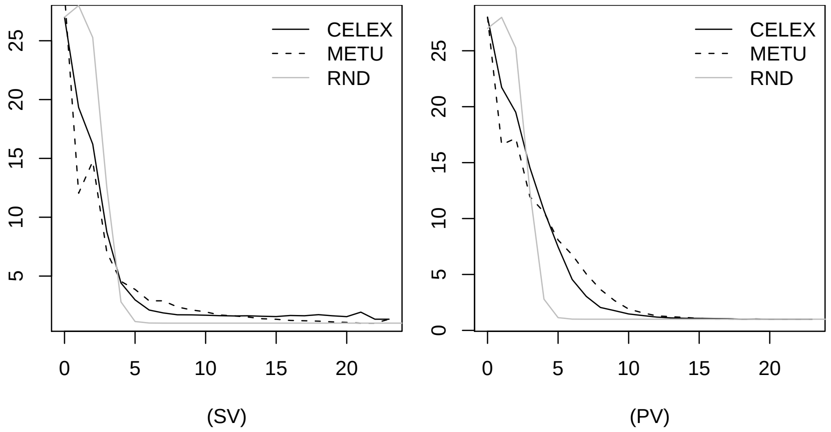


Figure 3: Mean SV and PV values for each word position. Indexes start from the beginning of the word for SV, and from the end of the word for PV.

values. The drop in the SV values is sharper. This is due to the fact that both languages in our study are primarily 'suffixing' languages: except a few productive linguistic prefixes, most of the productive morphological processes are due to the use of suffixes. This is clearly visible in the difference between the drop of the SV values from the beginning of the words to the end and the PV values from the end of the words to the beginning. Even though the languages differ in their morphological productivity, the graphs presented in Figure 3 do not reveal a big difference between the languages. However, we observe a clear difference between the behavior of SV and PV values.

The visualization in Figure 3 also clarifies the unexpected success of the SV thresholds tuned by Hafer and Weiss (1974) across different languages. The successful segmentation criteria they use depend on a combination of successor and predecessor values, where one of the thresholds is high and the other is low. As shown in Figure 3 the SV values tend to be one towards the end of a medium length word, likewise for the PV values towards the beginning. On the other hand, high SV scores can be found at the beginning, and high PV scores can be found at the end of the words. Hence, a high SV threshold in combination with a low (but greater than one) PV threshold translates to 'any PV value greater than one towards the beginning of a word'. And the reverse condition —high PV threshold, low SV threshold— detects any SV value greater than one towards the end of the words.

5 Normalization

The tendency of average SV and PV values dropping exponentially in Figure 3 can partially be explained by a very general process. Any process generating a large

letter		**r**		**e**		**a**		**d**		**i**		**n**		**g**	
SV	27		16		27		10		10		3		2		4
PV	7		5		16		10		25		5		13		28
SV/AVG	1.00		0.83		1.67		1.15		2.27		0.68		0.95		2.15
PV/AVG	2.44		1.14		2.18		0.94		1.73		0.26		0.70		1.00

Figure 4: Normalized SV and PV values.

number of word-like strings from a fixed alphabet would generate similar successor and predecessor counts. To demonstrate this, we will consider a process that creates random word like units, and compare it to actual natural language words. The light-gray line in Figure 3 shows the SV and PV values for such a process. We have generated random word-like units from an alphabet of size 30 (approximately the alphabet size for both of our word-lists), and generated $114,184$ (size of the CELEX word list) random 'words'.[4] Even though the random words were not formed by concatenating morphemes, they show the same tendency: At the beginning of the words the SV is high, and at the end of the words the PV is high. Naturally, the drop of the SV values of the random process does not show any differences from the PV values.

The main difference between real language data and the randomly generated data is that, rather than being a random collection of letters, the words in the real language data are formed by concatenating more basic units, morphemes. However, we observe the exponential drop of the SV and PV values for both the randomly generated data and the real language words. Hence, removing the effect of the letter concatenation process from the SV and PV values calculated for the real data should reveal the underlying process of morpheme concatenation better. To do that, we will follow a simple method: we will divide each variety value by the expected value in that position.[5]

Along with the previous SV and PV values presented in Figure 2, Figure 4 presents the normalized scores. The first difference to note is that both normalized scores for successor and predecessor values peak after `read-`. Since an SV value of 10 is closer to the expected value in position 3 (after `rea-`) than in position 4 (after `read-`) the scores are more sensitive to real boundaries. Of course, this may also increase the number of potential false positives. However, the results we present in Section 6 show that the normalization is indeed beneficial for increasing

[4] During the random word generation, letters are sampled similar to the letter distribution in the CELEX word list, and the word length distribution has been estimated from the combined 11 languages from the Europarl corpus. While providing a language-neutral word length distribution, this results in relatively long words compared to the more complete/balanced CELEX and METU word lists.

[5] Subtracting the expected SV/PV values from the calculated ones is another, arguably more intuitive, approach to normalize the SV values. The values obtained by subtracting the expected value from the calculated values have a similar distribution with the log of the values obtained by the normalization by division. The subtraction method does not require further log transformation, and it leads to higher accuracy in some of the conditions we tested. However, the subtraction method is more prone to changes in the size of the word list. The main reason for choosing to normalize by dividing is to get properly scaled values for the (semi-)supervised experiments discussed in the next section.

the segmentation performance.

The performance increase will be more apparent with the empirical tests. However, we can see another benefit of the normalization by looking at our example, and by visualizing the data. Intuitively, a normalized score of one means that the variety score is the same as the expected value. A score less than one is below the expected value, and a score greater than one is higher than the expected value. Hence, regardless of the position in the word, if the value is less than or around one there is no reason to get surprised, and no need to posit a boundary. However, if the value is significantly greater than one, it is more likely to be a boundary. And if we we take an even closer look at the data, we can see that the normalized variety values are approximately distributed according to a log-normal distribution. As a result if we take the normalization one step further, and get the logarithm of the values, we end up working with approximately normally distributed values. Figure 5 presents plots of density estimates of normalized and unnormalized SV and PV scores for both boundary (black lines) and non-boundary (gray lines) positions for English and Turkish word lists. In all cases, the modes of the distributions of boundaries are greater than the modes of the non-boundaries. However, Figure 5 also demonstrates that, for the SV values, the overlap between boundary and non-boundary values is clearly higher for the non-normalized case, and separation is not as clear as in the normalized case. Unfortunately, we do not see the same positive effect of normalization on the PV values. This is because of the fact that both languages are primarily suffixing languages and the suffixes tend to be shorter than the stems. The PV values at the end of the words are higher because of the process of morpheme concatenation as well. As a result, it is not possible to see if the high number of PV values at the end of a word is because of a genuine morpheme boundary, or because of the letter concatenation process. Since the number of genuine morpheme boundaries is lower at the beginning of the words, the SV values do not suffer from this phenomenon in either language we studied.

For both the SV and PV, the log-normalized values are (approximately) normally distributed. By estimating the parameters of the normal distributions for boundaries and non-boundaries, we can easily come up with strategies based on the SV that best suit the application at hand. A neutral method would be to view the segmentation task as binary classification of the data coming from two Gaussian distributions. Similarly, a more conservative estimate can be obtained by segmenting at values that are at the right tail of the combined normal distribution. For ease of comparison, in our experiments we will tune a threshold value that generates the best F_1-score for both normalized and unnormalized cases.

Another point to note from the visualization of the data in Figure 5 is that even though the unnormalized distributions for different languages seem to be rather different, the normalization takes away this dissimilarity. This may allow us to use strategies that are relatively language neutral. That is, by using normalized scores, the same model may work better cross-linguistically.

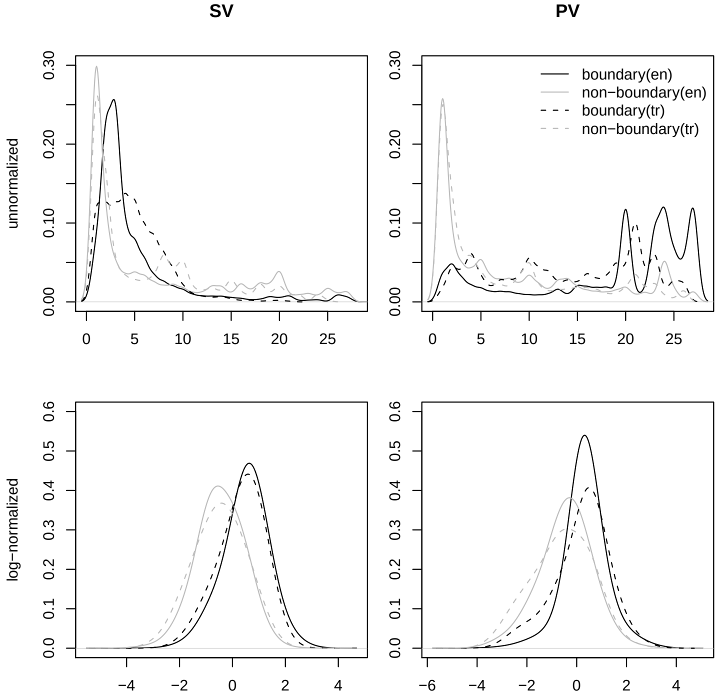


Figure 5: Density estimates for the SV and PV values with or without normalization.

6 Experiments

The most common application of using successor values for morphological segmentation has been in unsupervised morphological segmentation and analysis tasks. However, in all the examples in the literature that we are aware of, the method depends on thresholds tuned on a specific word list or intuitively by the designers of the algorithms. Arguably, the completely unsupervised use of the method is using changes (peaks) in the value of SV and PV as a criterion for segmentation. However, a small threshold is generally useful to guard against over-segmentation, and the methods deciding on the basis of a threshold tend to perform better. Another approach for completely unsupervised learning is based on taking the normalized SV and PV values as coming from two different (normal) distributions, and inducing the parameters of these distributions using an unsupervised method. Although complete unsupervised methods are worth exploring for their own sake, the methods we present below provide a better comparison between the normalized and unnormalized values, as well as comparing them to the previous results found in the literature.

To be able to demonstrate the effects of the normalization on the performance of the segmentation task, we present two sets of experiments in this section. First, we will present the *precision*, *recall* and F_1*-score* values for the segmentation of English and Turkish data sets using the SV and PV values individually, using a peak criterion on SV or PV values (SP an PP) and simple logical *and* and *or* combination of these values. In the second set of experiments, we will train a simple linear classifier.

In the first set of experiments we found threshold values that produce the best average F_1-score for 10-fold cross validation on our data sets. First we used thresholds for the individual SV and PV values. Second, we used the 'peak criterion', where we assumed a boundary where the value shows an increase. And last, we used simple logical *and* and *or* combinations of the SV and PV values.

Table 1.1 presents these results, along with two baselines. The line marked as 'Letters' presents the simple strategy of segmenting words to single letters. The line marked as 'Morfessor' presents the results obtained using the Morfessor 1.0 baseline (Creutz and Lagus 2005). As expected, by using the normalized SV values, we can find thresholds that perform better. However, the normalization for PV values produce even worse than non-normalized values, which also took F_1-scores for some of the combinations down. This can be corrected by adding another criterion by favoring boundaries at the end of the words. This is indeed useful for improving the performance of the normalized PV based segmentation criteria in Table 1.1. This type of correction is rather ad hoc and introduces additional parameters. For the simple experiments, we will not report the results of such an ad hoc correction here. However, the effect of using the information on the position in the word is demonstrated in the second set of experiments we report below.

We did a large number of tests and presented in Table 1.1 to get an insight into the method, and to be able to compare the effectiveness of the measures. However, tuning thresholds and combining multiple indicators in a sensible way is, at best,

		English			Turkish		
	Method	P	R	F_1	P	R	F_1
Letters		0.19	1.00	0.32	0.22	1.00	0.36
Morfessor		0.82	0.55	0.66	0.77	0.50	0.60
unnormalized	SV	0.25	0.93	0.40	0.34	0.74	0.47
	PV	0.51	0.65	0.57	0.42	0.78	0.55
	SP	0.46	0.59	0.52	0.47	0.49	0.48
	PP	0.54	0.52	0.53	0.37	0.38	0.38
	SV and PV	0.65	0.69	0.67	0.62	0.66	0.64
	SV or PV	0.40	0.67	0.51	0.38	0.90	0.53
normalized	SV	0.42	0.72	0.53	0.45	0.74	0.56
	PV	0.34	0.88	0.49	0.44	0.68	0.53
	SP	0.53	0.59	0.56	0.37	0.78	0.50
	PP	0.36	0.60	0.45	0.31	0.75	0.44
	SV and PV	0.61	0.70	0.65	0.67	0.57	0.62
	SV or PV	0.42	0.73	0.53	0.45	0.74	0.56

Table 1.1: *P*recision/*R*ecall/*F*-score values optimized for best F-score. The first block presents two baselines.

a cumbersome task. A better way to make use of these values is using them as features in a (semi-)supervised learning method. The second set of experiments we conducted is based on training a simple linear classifier (Fan et al. 2008) using the normalized and unnormalized SV and PV values. As well as the SV and PV values, we used two other sets of features. First, two additional binary features that indicate the occurrence of the suffix and the prefix (in respect to the candidate boundary) as a separate word in the word list. Second, two index numbers corresponding to the offsets of the candidate boundary from the beginning and end of the word. The standalone occurrence of suffix and prefix in the word list is frequently used by the other models in the literature. The addition of index features, on the other hand, allows the learner to correct the problem with the normalized PV values that we observed above.

The results of the supervised learning experiments are presented in Table 1.2. Each row in the table presents the precision, recall and F_1-Scores for different combination of the features. All values are averages of 10-fold cross validation on the CELEX and METU word lists. The experiments reported at the rows marked with '+index' have been generated using two integer features representing the index from the beginning and the end of the word. The rows marked '+w' report the results from the experiments where we added two additional binary features indicating the existence of the suffix and the prefix as a standalone word in the word list. Except the 'PV' and 'SV+PV' features for English, all combinations of the features show a consistent increase of F_1-score for both languages. And as expected, the addition of '+index' also allows the model to generalize better using

		English			Turkish		
	Method	P	R	F_1	P	R	F_1
unnormalized	SV	0.51	0.35	0.41	0.48	0.55	0.51
	PV	0.29	0.64	0.40	0.27	0.91	0.42
	SV+PV	0.40	0.84	0.54	0.47	0.71	0.57
	SV+PV+index	0.40	0.89	0.55	0.40	0.85	0.55
	SV+PV+index+w	0.46	0.83	0.59	0.46	0.87	0.60
normalized	SV	0.36	0.83	0.50	0.45	0.75	0.56
	PV	0.30	0.69	0.42	0.45	0.69	0.55
	SV+PV	0.39	0.85	0.53	0.56	0.76	0.64
	SV+PV+index	0.43	0.83	0.57	0.59	0.80	0.68
	SV+PV+index+w	0.60	0.83	0.69	0.62	0.81	0.70

Table 1.2: Results of supervised learning.

SV and PV values.

Encouraged by the similarity of normalized distributions for two languages in Figure 5, we conducted four more experiments: we trained the classifier using the full set of features on one of the languages, and tested on the other. As expected, the models trained by the unnormalized data performed poorly. Training on Turkish data and testing on English data resulted in an F_1-score of 0.34, barely above the 'Letters' baseline. Training the system with English and testing on Turkish did even worse, an F_1-score of 0.23. However, the F_1-scores using normalized scores for the same setup resulted in F_1-scores of 0.67 and 0.64 respectively—better than any of the hand-tuned thresholds in Table 1.1.

7 Discussion and Conclusions

Along with an in-depth analysis, we presented in this paper a method to improve successor variety, an old but frequently used measure for segmentation. The measure we discussed in this paper, successor variety, shares a principle with a few other measures used in the literature: the predictability within the units (e.g., morphemes) is high, predictability between the units is low. The analysis presented here focuses on the application of the successor variety to segment words into morphemes. However, the improvement suggested in this paper can be used in other segmentation applications, and can be applied to other measures, e.g., entropy, if used in a similar fashion.

The normalization idea presented here is based on the observation that, even for a random letter concatenation process, the successor values tend to be high at the beginning, and drop exponentially as we increase the prefix length. This suggests that if we can isolate the concatenation at a higher level—in our case concatenation of the morphemes— from the letter concatenation process, we can increase the efficiency and arrive at a better interpretation of the relation between the measure and the boundaries. The normalization method we presented in this paper achieves that by simply dividing the calculated successor value to the expected value after an equal prefix length. Alternative normalization methods are possible, for example simply subtracting the expected value from the calculated score.

While calculating the expected value, our method only considers the length of the prefix (the expected SV is higher for shorter prefixes). However, natural languages have a number of other regularities that affect the SV values. Particularly, not every letter or letter class is likely to be followed by equal number of successor letters. For example, the languages we consider have more consonants than vowels, and typically, consonants are followed by vowels and vowels are followed by consonants. This results in vowels in average to have higher successor values than consonants. Arguably, incorporating this information while calculating the expected values may provide a better normalization. In a number of preliminary experiments that we did not report in this paper, we could not find any consistent improvement by incorporating this information in the normalization process.

The experiments we conducted in two different languages demonstrate that the normalization method proposed here is effective in increasing the performance of the segmentation methods based on successor variety. The effect seems to be more useful for the SV values, however, with correct use of other cues, we also demonstrated that it may increase the effectiveness of the PV values as well.

References

Al-Shalabi, Riyad, Ghassan Kannan, Iyad Hilat, Ahmad Ababneh, and Ahmad Al-Zubi (2005), Experiments with the successor variety algorithm using the cutoff and entropy methods, *Information Technology Journal.*

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

Bordag, Stefan (2005), Unsupervised knowledge-free morpheme boundary detection., *The Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP).*

Bordag, Stefan (2007), Unsupervised and knowledge-free morpheme segmentation and analysis, *The Working Notes for the CLEF Workshop 2007.*

Brent, Michael R. (1996), Advances in the computational study of language acquisition, *Cognition* **61**, pp. 1–38.

Brent, Michael R. (1999), An efficient, probabilistically sound algorithm for segmentation and word discovery, *Machine Learning* **34** (1–3), pp. 71–105.

Cohen, Paul, Niall Adams, and Brent Heeringa (2007), Voting experts: An unsupervised algorithm for segmenting sequences, *Intelligent Data Analysis* **11** (6), pp. 607–625.

Çöltekin, Çağrı (2010), A freely available morphological analyzer for turkish, *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC2010).*

Creutz, Mathias and Krista Lagus (2005), Inducing the morphological lexicon of a natural language from unannotated text, *Proceedings of the International* *and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05)*, Espoo, Finland, pp. 106–113.

Creutz, Mathias and Krista Lagus (2007), Unsupervised models for morpheme segmentation and morphology learning, *ACM Trans. Speech Lang. Process.* **4** (1), pp. 3, ACM, New York, NY, USA.

Creutz, Mathias and Krister Lindén (2004), Morpheme segmentation gold standards for Finnish and English., *Publications in Computer and Information Science A77*, Helsinki University of Technology.

Déjean, Hervé (1998), Morphemes as necessary concept for structures discovery from untagged corpora, *Workshop on Paradigms and Grounding in Natural Language Learning*, pp. 295–299.

Demberg, Vera (2007), A language-independent unsupervised model for morphological segmentation, *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Association for Computational Linguistics, Prague, Czech Republic, pp. 920–927.

Fan, R.-E., K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin (2008), Liblinear: A library for large linear classification, *Journal of Machine Learning Research* (9), pp. 1871–1874.

Goldsmith, John (2001), Unsupervised learning of the morphology of a natural language, *Computational Linguistics* **27** (2), pp. 153–198, MIT Press, Cambridge, MA, USA.

Goldsmith, John (2006), An algorithm for the unsupervised learning of morphology, *Natural Language Engineering* **12** (04), pp. 353–371.

Goldwater, Sharon, Thomas L. Griffiths, and Mark Johnson (2009), A Bayesian framework for word segmentation: Exploring the effects of context, *Cognition* **112**, pp. 21–54.

Hafer, Margaret A. and Stephen F. Weiss (1974), Word segmentation by letter successor varieties, *Information Storage and Retrieval* **10** (11–12), pp. 371–385.

Harris, Zellig S. (1955), From phoneme to morpheme, *Language* **31** (2), pp. 190–222, Linguistic Society of America.

Saffran, Jenny R., Richard N. Aslin, and Elissa L. Newport (1996), Statistical learning by 8-month old infants, *Science* **274** (5294), pp. 1926–1928.

Say, Bilge, Deniz Zeyrek, Kemal Oflazer, and Umut Özge (2002), Development of a corpus and a treebank for present-day written Turkish, *Proceedings of the Eleventh International Conference of Turkish Linguistics*.

Stein, Benno and Martin Potthast (2008), Putting successor variety stemming to work, *in* Decker, Reinhold and Hans J. Lenz, editors, *Advances in Data Analysis*, Springer, pp. 367–374.