Unsupervised Segmentation of Words Using Prior Distributions of Morph Length and Frequency

Mathias Creutz
Neural Networks Research Centre, Helsinki University of Technology
P.O.Box 9800, FIN-02015 HUT, Finland
Mathias.Creutz@hut.fi

Abstract

We present a language-independent and unsupervised algorithm for the segmentation of words into morphs. The algorithm is based on a new generative probabilistic model, which makes use of relevant prior information on the length and frequency distributions of morphs in a language. Our algorithm is shown to outperform two competing algorithms, when evaluated on data from a language with agglutinative morphology (Finnish), and to perform well also on English data.

1 Introduction

In order to artificially "understand" or produce natural language, a system presumably has to know the elementary building blocks, i.e., the lexicon, of the language. Additionally, the system needs to model the relations between these lexical units. Many existing NLP (natural language processing) applications make use of *words* as such units. For instance, in statistical language modelling, probabilities of word sequences are typically estimated, and bag-of-word models are common in information retrieval.

However, for some languages it is infeasible to construct lexicons for NLP applications, if the lexicons contain entire words. In especially agglutinative languages,[1] such as Finnish and Turkish, the number of possible different word forms is simply too high. For example, in Finnish, a single verb may appear in thousands of different forms (Karlsson, 1987).

According to linguistic theory, words are built from smaller units, morphemes. Morphemes are the smallest meaning-bearing elements of language and could be used as lexical units instead of entire words. However, the construction of a comprehensive morphological lexicon or analyzer based on linguistic theory requires a considerable amount of work by experts. This is both time-consuming and expensive and hardly applicable to all languages. Furthermore, as language evolves the lexicon must be updated continuously in order to remain up-to-date.

Alternatively, an interesting field of research lies open: Minimally supervised algorithms can be designed that automatically discover morphemes or morpheme-like units from data. There exist a number of such algorithms, some of which are entirely unsupervised and others that use some knowledge of the language. In the following, we discuss recent unsupervised algorithms and refer the reader to (Goldsmith, 2001) for a comprehensive survey of previous research in the whole field.

Many algorithms proceed by segmenting (i.e., splitting) words into smaller components. Often the limiting assumption is made that words consist of only one stem followed by one (possibly empty) suffix (Déjean, 1998; Snover and Brent, 2001; Snover et al., 2002). This limitation is reduced in (Goldsmith, 2001) by allowing a recursive structure, where stems can have inner structure, so that they in turn consist of a substem and a suffix. Also

[1] In agglutinative languages words are formed by the concatenation of morphemes.

Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, July 2003, pp. 280-287.

prefixes are possible. However, for languages with agglutinative morphology this may not be enough. In Finnish, a word can consist of lengthy sequences of alternating stems and affixes.

Some morphology discovery algorithms learn relationships between words by comparing the orthographic or semantic similarity of the words (Schone and Jurafsky, 2000; Neuvel and Fulop, 2002; Baroni et al., 2002). Here a small number of components per word are assumed, which makes the approaches difficult to apply as such to agglutinative languages.

We previously presented two segmentation algorithms suitable for agglutinative languages (Creutz and Lagus, 2002). The algorithms learn a set of segments, which we call *morphs*, from a corpus. Stems and affixes are not distinguished as separate categories by the algorithms, and in that sense they resemble algorithms for text segmentation and word discovery, such as (Deligne and Bimbot, 1997; Brent, 1999; Kit and Wilks, 1999; Yu, 2000). However, we observed that for the corpus size studied (100 000 words), our two algorithms were somewhat prone to excessive segmentation of words.

In this paper, we aim at overcoming the problem of excessive segmentation, particularly when small corpora (up to 200 000 words) are used for training. We present a new segmentation algorithm, which is language independent and works in an unsupervised fashion. Since the results obtained suggest that the algorithm performs rather well, it could possibly be suitable for languages for which only small amounts of written text are available.

The model is formulated in a probabilistic Bayesian framework. It makes use of explicit prior information in the form of probability distributions for morph length and morph frequency. The model is based on the same kind of reasoning as the probabilistic model in (Brent, 1999). While Brent's model displays a prior probability that exponentially decreases with word length (with one character as the most common length), our model uses a probability distribution that more accurately models the real length distribution. Also Brent's frequency distribution differs from ours, which we derive from Mandelbrot's correction of Zipf's law (cf. Section 2.5).

Our model requires that the values of two parameters be set: (i) our prior belief of the *most common morph length*, and (ii) our prior belief of the *proportion of morph types*[2] *that occur only once in the corpus.* These morph types are called *hapax legomena*. While the former is a rather intuitive measure, the latter may not appear as intuitive. However, the proportion of hapax legomena may be interpreted as a measure of the richness of the text. Also note that since the most common morph length is calculated for morph types, not tokens, it is not independent of the corpus size. A larger corpus usually requires a higher average morph length, a fact that is stated for word lengths in (Baayen, 2001).

As an evaluation criterion for the performance of our method and two reference methods we use a measure that reflects the ability to recognize real morphemes of the language by examining the morphs found by the algorithm.

2 Probabilistic generative model

In this section we derive the new model. We follow a step-by-step process, during which a morph lexicon and a corpus are generated. The morphs in the lexicon are strings that emerge as a result of a stochastic process. The corpus is formed through another stochastic process that picks morphs from the lexicon and places them in a sequence. At two points of the process, prior knowledge is required in the form of two real numbers: the most common morph length and the proportion of hapax legomena morphs.

The model can be used for segmentation of words by requiring that the corpus created is exactly the input data. By selecting the most probable morph lexicon that can produce the input data, we obtain a segmentation of the words in the corpus, since we can rewrite every word as a sequence of morphs.

2.1 Size of the morph lexicon

We start the generation process by deciding the number of morphs in the morph lexicon (type count). This number is denoted by n_μ and its probability $p(n_\mu)$ follows the uniform distribution. This means that, a priori, no lexicon size is more probable than another.[3]

[2] We use standard terminology: Morph *types* are the set of different, distinct morphs. By contrast, morph *tokens* are the instances (or occurrences) of morphs in the corpus.

[3] This is an improper prior, but it is of little practical significance for two reasons: (i) This stage of the generation process

2.2 Morph lengths

For each morph in the lexicon, we independently choose its length in characters according to the gamma distribution:

$$p(l_{\mu_i}) = \frac{1}{\Gamma(\alpha)\beta^\alpha} {l_{\mu_i}}^{\alpha-1} e^{-l_{\mu_i}/\beta}, \quad (1)$$

where l_{μ_i} is the length in characters of the ith morph, and α and β are constants. $\Gamma(\alpha)$ is the gamma function:

$$\Gamma(\alpha) = \int_0^\infty z^{\alpha-1} e^{-z} dz. \quad (2)$$

The maximum value of the density occurs at $l_{\mu_i} = (\alpha - 1)\beta$, which corresponds to the most common morph length in the lexicon. When β is set to one, and α to one plus our prior belief of the most common morph length, the pdf (probability density function) is completely defined.

We have chosen the gamma distribution for morph lengths, because it corresponds rather well to the real length distribution observed for word types in Finnish and English corpora that we have studied. The distribution also fits the length distribution of the morpheme labels used as a reference (cf. Section 3). A Poisson distribution can be justified and has been used in order to model the length distribution of word and morph *tokens* [e.g., (Creutz and Lagus, 2002)], but for morph *types* we have chosen the gamma distribution, which has a thicker tail.

2.3 Morph strings

For each morph μ_i, we decide the character string it consists of: We independently choose l_{μ_i} characters at random from the alphabet in use. The probability of each character c_j is the maximum likelihood estimate of the occurrence of this character in the corpus:[4]

$$p(c_j) = \frac{n_{c_j}}{\sum_k n_{c_k}}, \quad (3)$$

where n_{c_j} is the number of occurrences of the character c_j in the corpus, and $\sum_k n_{c_k}$ is the total number of characters in the corpus.

only contributes with one probability value, which will have a negligible effect on the model as a whole. (ii) A proper probability density function would presumably be very flat, which would hardly help guiding the search towards an optimal model.

[4]Alternatively, the maximum likelihood estimate of the occurrence of the character in the *lexicon* could be used.

2.4 Morph order in the lexicon

The lexicon consists of a set of n_μ morphs and it makes no difference in which order these morphs have emerged. Regardless of their initial order, the morphs can be sorted into a uniquely defined (e.g., alphabetical) order. Since there are $n_\mu!$ ways to order n_μ different elements,[5] we multiply the probability accumulated so far by $n_\mu!$:

$$p(\text{lexicon}) = p(n_\mu) \prod_{i=1}^{n_\mu} \Big[p(l_{\mu_i}) \prod_{j=1}^{l_{\mu_i}} p(c_j) \Big] \cdot n_\mu! \quad (4)$$

2.5 Morph frequencies

The next step is to generate a corpus using the morph lexicon obtained in the previous steps. First, we independently choose the number of times each morph occurs in the corpus. We pursue the following line of thought:

Zipf has studied the relationship between the frequency of a word, f, and its rank, z.[6] He suggests that the frequency of a word is inversely proportional to its rank. Mandelbrot has refined Zipf's formula, and suggests a more general relationship [see, e.g., (Baayen, 2001)]:

$$f = C(z + b)^{-a}, \quad (5)$$

where C, a and b are parameters of a text.

Let us derive a probability distribution from Mandelbrot's formula. The rank of a word as a function of its frequency can be obtained by solving for z from (5):

$$z = C^{\frac{1}{a}} f^{-\frac{1}{a}} - b. \quad (6)$$

Suppose that one wants to know the number of words that have a frequency close to f rather than the rank of the word with frequency f. In order to obtain this information, we choose an arbitrary interval around f: $[(1/\gamma)f \ldots \gamma f[$, where $\gamma > 1$, and compute the rank at the endpoints of the interval. The difference is an estimate of the number of words

[5]Strictly speaking, our probabilistic model is not perfect, since we do not make sure that no morph can appear more than once in the lexicon.

[6]The rank of a word is the position of the word in a list, where the words have been sorted according to falling frequency.

that fall within the interval, i.e., have a frequency close to f:

$$n_f = z_{1/\gamma} - z_\gamma = (\gamma^{\frac{1}{a}} - \gamma^{-\frac{1}{a}})C^{\frac{1}{a}} f^{-\frac{1}{a}}. \quad (7)$$

This can be transformed into an exponential pdf by (i) binning the frequency axis so that there are no overlapping intervals. (This means that the frequency axis is divided into non-overlapping intervals $[(1/\gamma)\hat{f} \ldots \gamma\hat{f}[$, which is equivalent to having $\hat{f}$ values that are powers of γ^2: $\hat{f}_0 = \gamma^0 = 1, \hat{f}_1 = \gamma^2, \hat{f}_2 = \gamma^4, \ldots$ All frequencies f are rounded to the closest $\hat{f}$.) Next (ii), we normalize the number of words with a frequency close to $\hat{f}$ with the total number of words $\sum_{\hat{f}} n_{\hat{f}}$. Furthermore (iii), $\hat{f}$ is written as $e^{\log \hat{f}}$, and (iv) C must be chosen so that the normalization coefficient equals $1/a$, which yields a proper pdf that integrates to one. Note also the factor $\log \gamma^2$. Like $\hat{f}$, $\log \hat{f}$ is a discrete variable. We approximate the integral of the density function around each value $\log \hat{f}$ by multiplying with the difference between two successive $\log \hat{f}$ values, which equals $\log \gamma^2$:

$$\begin{aligned} p(f \in [(1/\gamma)\hat{f} \ldots \gamma\hat{f}[) &= \frac{\gamma^{\frac{1}{a}} - \gamma^{-\frac{1}{a}}}{\sum_{\hat{f}} n_{\hat{f}}} C^{\frac{1}{a}} e^{-\frac{1}{a}\log \hat{f}} \\ &= \frac{1}{a} e^{-\frac{1}{a}\log \hat{f}} \cdot \log \gamma^2. \quad (8) \end{aligned}$$

Now, if we assume that Zipf's and Madelbrot's formulae apply to morphs as well as to words, we can use formula (8) for every morph frequency f_{μ_i}, which is the number of occurrences (or frequency) of the morph μ_i in the corpus (token count). However, values for a and γ^2 must be chosen. We set γ^2 to 1.59, which is the lowest value for which no empty frequency bins will appear.[7] For $f_{\mu_i} = 1$, (8) reduces to $\log \gamma^2 / a$. We set this value equal to our prior belief of the proportion of morph types that are to occur only once in the corpus (hapax legomena).

2.6 Corpus

The morphs and their frequencies have been set. The order of the morphs in the corpus remains to be decided. The probability of one particular order is the inverse of the multinomial:

$$p(\text{corpus}) = \Big(\frac{(\sum_{i=1}^{n_\mu} f_{\mu_i})!}{\prod_{i=1}^{n_\mu} f_{\mu_i}!}\Big)^{-1} = \Big(\frac{N!}{\prod_{i=1}^{n_\mu} f_{\mu_i}!}\Big)^{-1}. \quad (9)$$

The numerator of the multinomial is the factorial of the total number of morph tokens, N, which equals the sum of frequencies of every morph type. The denominator is the product of the factorial of the frequency of each morph type.

2.7 Search for the optimal model

The search for the optimal model given our input data corresponds closely to the recursive segmentation algorithm presented in (Creutz and Lagus, 2002). The search takes place in batch mode, but could as well be done incrementally. All words in the data are randomly shuffled, and for each word, every split into two parts is tested. The most probable split location (or no split) is selected and in case of a split, the two parts are recursively split in two. All words are iteratively reprocessed until the probability of the model converges.

3 Evaluation

From the point of view of linguistic theory, it is possible to come up with different plausible suggestions for the correct location of morpheme boundaries. Some of the solutions may be more elegant than others,[8] but it is difficult to say if the most elegant scheme will work best in practice, when real NLP applications are concerned.

We utilize an evaluation method for segmentation of words presented in (Creutz and Lagus, 2002). In this method, segments are *not* compared to one single "correct" segmentation. The evaluation criterion can rather be interpreted from the point of view of language "understanding". A morph discovered by the segmentation algorithm is considered to be "understood", if there is a low-ambiguity mapping from the morph to a corresponding morpheme. Alternatively, a morph may correspond to a sequence of morphemes, if these morphemes are very likely to occur together. The idea is that if an entirely new word form is encountered, the system will "understand" it by decomposing it into morphs that it "understands". A segmentation algorithm that segments

[7] Empty bins can appear for small values of f_{μ_i} due to f_{μ_i}'s being rounded to the closest $\hat{f}_{\mu_i}$, which is a power of γ^2.

[8] Cf. "hop + ed" vs. "hope + d" (past tense of "to hope").

words into too small parts will perform poorly due to high ambiguity. At the other extreme, an algorithm that is reluctant at splitting words will have bad generalization ability to new word forms.

Reference morpheme sequences for the words are obtained using existing software for automatic morphological analysis based on the two-level morphology of Koskenniemi (1983). For each word form, the analyzer outputs the base form of the word together with grammatical tags. By filtering the output, we get a sequence of morpheme labels that appear in the correct order and represent correct morphemes rather closely. Note, however, that the morpheme labels are not necessarily orthographically similar to the morphemes they represent.

The exact procedure for evaluating the segmentation of a set of words consists of the following steps:

(1) Segment the words in the corpus using the automatic segmentation algorithm.

(2) Divide the segmented data into two parts of equal size. Collect all segmented word forms from the first part into a training vocabulary and collect all segmented word forms from the second part into a test vocabulary.

(3) Align the segmentation of the words in the training vocabulary with the corresponding reference morpheme label sequences. Each morph must be aligned with one or more consecutive morpheme labels and each morpheme label must be aligned with at least one morph; e.g., for a hypothetical segmentation of the English word *winners'*:

Morpheme labels	*win*		*-ER*	*PL*	*GEN*
Morph sequence	w	inn	er	s'	

(4) Estimate conditional probabilities for the morph/morpheme mappings computed over the whole training vocabulary: $p(\text{morpheme} \mid \text{morph})$. Re-align using the Viterbi algorithm and employ the Expectation-Maximization algorithm iteratively until convergence of the probabilities.

(5) The quality of the segmentation is evaluated on the test vocabulary. The segmented words in the test vocabulary are aligned against their reference morpheme label sequences according to the conditional probabilities learned from the training vocabulary. To measure the quality of the segmentation we compute the expectation of the proportion of correct mappings from morphs to morpheme labels, $E\{p(\text{morpheme} \mid \text{morph})\}$:

$$\frac{1}{N}\sum_{i=1}^{N} p_i(\text{morpheme} \mid \text{morph}), \qquad (10)$$

where N is the number of morph/morpheme mappings, and $p_i(\cdot)$ is the probability associated with the ith mapping. Thus, we measure the *proportion of morphemes in the test vocabulary that we can expect to recognize correctly* by examining the morph segments.[9]

4 Experiments

We have conducted experiments involving (i) three different segmentation algorithms, (ii) two corpora in different languages (Finnish and English), and (iii) data sizes ranging from 2000 words to 200 000 words.

4.1 Segmentation algorithms

The new probabilistic method is compared to two existing segmentation methods: the *Recursive MDL* method presented in (Creutz and Lagus, 2002)[10] and John Goldsmith's algorithm called *Linguistica* (Goldsmith, 2001).[11] Both methods use MDL (Minimum Description Length) (Rissanen, 1989) as a criterion for model optimization.

The effect of using prior information on the distribution of morph length and frequency can be assessed by comparing the probabilistic method to Recursive MDL, since both methods utilize the same search algorithm, but Recursive MDL does not make use of explicit prior information.

Furthermore, the possible benefit of using the two sources of prior information can be compared against the possible benefit of grouping stems and suffixes into signatures. The latter technique is employed by Linguistica.

4.2 Data

The Finnish data consists of subsets of a newspaper text corpus from CSC,[12] from which nonwords (numbers and punctuation marks) have been

[9] In (Creutz and Lagus, 2002) the results are reported less intuitively as the "alignment distance", i.e., the negative logprob of the entire test set: $-\log \prod p_i(\text{morpheme} \mid \text{morph})$.

[10] Online demo at http://www.cis.hut.fi/projects/morpho/.

[11] The software can be downloaded from http://humanities.uchicago.edu/faculty/goldsmith/Linguistica2000/.

[12] http://www.csc.fi/kielipankki/

removed. The reference morpheme labels have been filtered out from a morphosyntactic analysis of the text produced by the Connexor FDG parser.[13]

The English corpus consists of mainly newspaper text (with non-words removed) from the Brown corpus.[14] A morphological analysis of the words has been performed using the Lingsoft ENGTWOL analyzer.[15]

For both languages data sizes of 2000, 5000, 10 000, 50 000, 100 000, and 200 000 have been used. A notable difference between the morphological structure of the languages lies in the fact that whereas there are about 17 000 English word types in the largest data set, the corresponding number of Finnish word types is 58 000.

4.3 Parameters

In order to select good prior values for the probabilistic method, we have used separate development test sets that are independent of the final data sets. Morph length and morph frequency distributions have been computed for the reference morpheme representations of the development test sets. The prior values for most common morph length and proportion of hapax legomena have been adjusted in order to produce distributions that fit the reference as well as possible.

We thus assume that we can make a good guess of the final morph length and frequency distributions. Note, however, that our reference is an approximation of a *morpheme* representation. As the segmentation algorithms produce morphs, not morphemes, we can expect to obtain a larger number of morphs due to allomorphy. Note also that we do not optimize for segmentation performance on the development test set; we only choose the best fit for the morph length and frequency distributions.

As for the two other segmentation algorithms, Recursive MDL has no parameters to adjust. In Linguistica we have used *Method A Suffixes + Find prefixes from stems* with other parameters left at their default values. We are unaware whether another configuration could be more advantageous for Linguistica.

[13] http://www.connexor.fi/

[14] The Brown corpus is available at the Linguistic Data Consortium at http://www.ldc.upenn.edu/.

[15] http://www.lingsoft.fi/

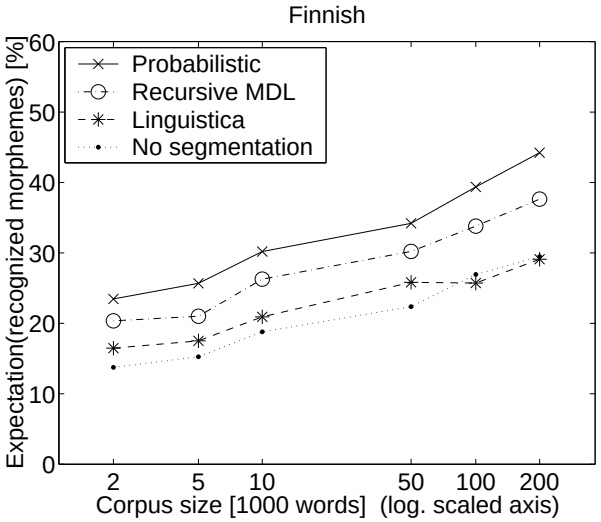

Figure 1: Expectation of the percentage of recognized morphemes for Finnish data.

4.4 Results

The expected proportion of morphemes recognized by the three segmentation methods are plotted in Figures 1 and 2 for different sizes of the Finnish and English corpora. The search algorithm used in the probabilistic method and Recursive MDL involve randomness and therefore every value shown for these two methods is the average obtained over ten runs with different random seeds. However, the fluctuations due to random behaviour are very small and paired t-tests show significant differences at the significance level of 0.01 for all pair-wise comparisons of the methods at all corpus sizes.

For Finnish, all methods show a curve that mainly increases as a function of the corpus size. The probabilistic method is the best with morpheme recognition percentages between 23.5% and 44.2%. Linguistica performs worst with percentages between 16.5% and 29.1%. None of the methods are close to ideal performance, which, however, is lower than 100%. This is due to the fact that the test vocabulary contains a number of morphemes that are not present in the training vocabulary, and thus are impossible to recognize. The proportion of unrecognizable morphemes is highest for the smallest corpus size (32.5%) and decreases to 8.8% for the largest corpus size.

The evaluation measure used unfortunately scores

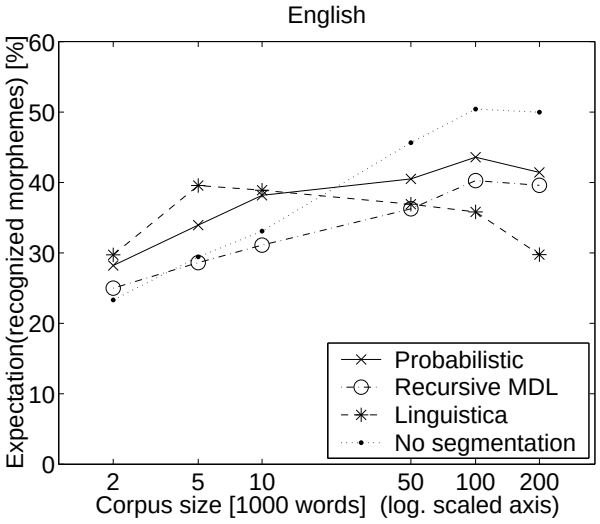

Figure 2: Expectation of the percentage of recognized morphemes for English data.

a baseline of no segmentation fairly high. The no-segmentation baseline corresponds to a system that recognizes the training vocabulary fully, but has no ability to generalize to any other word form.

The results for English are different. Linguistica is the best method for corpus sizes below 50 000 words, but its performance degrades from the maximum of 39.6% at 10 000 words to 29.8% for the largest data set. The probabilistic method is constantly better than Recursive MDL and both methods outperform Linguistica beyond 50 000 words. The recognition percentages of the probabilistic method vary between 28.2% and 43.6%. However, for corpus sizes above 10 000 words none of the three methods outperform the no-segmentation baseline.

Overall, the results for English are closer to ideal performance than was the case for Finnish. This is partly due to the fact that the proportion of unseen morphemes that are impossible to recognize is higher for English (44.5% at 2000 words, 19.0% at 200 000 words).

As far as the time consumption of the algorithms is concerned, the largest Finnish corpus took 20 minutes to process for the probabilistic method and Recursive MDL, and 40 minutes for Linguistica. The largest English corpus was processed in less than three minutes by all the algorithms. The tests were run on a 900 MHz AMD Duron processor with 256 MB RAM.

5 Discussion

For small data sizes, Recursive MDL has a tendency to split words into too small segments, whereas Linguistica is much more reluctant at splitting words, due to its use of signatures. The extent to which the probabilistic method splits words lies somewhere in between the two other methods.

Our evaluation measure favours low ambiguity as long as the ability to generalize to new word forms does not suffer. This works against all segmentation methods for English at larger data sizes. The English language has rather simple morphology, which means that the number of different possible word forms is limited. The larger the training vocabulary, the broader coverage of the test vocabulary, and therefore the no-segmentation approach works surprisingly well. Segmentation always increases ambiguity, which especially Linguistica suffers from as it discovers more and more signatures and short suffixes as the amount of data increases. For instance, a final 's' stripped off its stem can be either a noun or a verb ending, and a final 'e' is very ambiguous, as it belongs to orthography rather than morphology and does not correspond to any morpheme.

Finnish morphology is more complex and there are endless possibilities to construct new word forms. As can be seen from Figure 1, the probabilistic method and Recursive MDL perform better than the no-segmentation baseline for all data sizes.

The segmentations could be evaluated using other measures, but for language modelling purposes, we believe that the evaluation measure should not favour shattering of very common strings, even though they correspond to more than one morpheme. These strings should rather work as individual vocabulary items in the model. It has been shown that increased performance of n-gram models can be obtained by adding larger units consisting of common word sequences to the vocabulary; see e.g., (Deligne and Bimbot, 1995). Nevertheless, in the near future we wish to explore possibilities of using complementary and more standard evaluation measures, such as precision, recall, and F-measure of the discovered morph boundaries.

Concerning the length and frequency prior distributions in the probabilistic model, one notes that they are very general and do not make far-reaching

assumptions about the behaviour of natural language. In fact, Zipf's law has been shown to apply to randomly generated artificial texts (Li, 1992). In our implementation, due to the independence assumptions made in the model and due to the search algorithm used, the choice of a prior value for the most common morph length is more important than the hapax legomena value. If a very bad prior value for the most common morph length is used performance drops by twelve percentage units, whereas extreme hapax legomena values only reduces performance by two percentage units. But note that the two values are dependent: A greater average morph length means a greater number of hapax legomena and vice versa.

There is always room for improvement. Our current model does not represent contextual dependencies, such as phonological rules or morphotactic limitations on morph order. Nor does it identify which morphs are allomorphs of the same morpheme, e.g., "*city*" and "*citi* + es". In the future, we expect to address these problems by using statistical language modelling techniques. We will also study how the algorithms scale to considerably larger corpora.

6 Conclusions

The results we have obtained suggest that the performance of a segmentation algorithm can indeed be increased by using prior information of general nature, when this information is expressed mathematically as part of a probabilistic model. Furthermore, we have reasons to believe that the morph segments obtained can be useful as components of a statistical language model.

Acknowledgements

I am most grateful to Krista Lagus, Krister Lindén, and Anders Ahlbäck, as well as the anonymous reviewers for their valuable comments.

References

R. H. Baayen. 2001. *Word Frequency Distributions*. Kluwer Academic Publishers.

M. Baroni, J. Matiasek, and H. Trost. 2002. Unsupervised learning of morphologically related words based on orthographic and semantic similarity. In *Proc. ACL Workshop Morphol. & Phonol. Learning*, pp. 48–57.

M. R. Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105.

M. Creutz and K. Lagus. 2002. Unsupervised discovery of morphemes. In *Proc. ACL Workshop on Morphol. and Phonological Learning*, pp. 21–30, Philadelphia.

H. Déjean. 1998. Morphemes as necessary concept for structures discovery from untagged corpora. In *Workshop on Paradigms and Grounding in Nat. Lang. Learning*, pp. 295–299, Adelaide.

S. Deligne and F. Bimbot. 1995. Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams. In *Proc. ICASSP*.

S. Deligne and F. Bimbot. 1997. Inference of variable-length linguistic and acoustic units by multigrams. *Speech Communication*, 23:223–241.

J. Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198.

F. Karlsson. 1987. *Finnish Grammar*. WSOY, 2nd ed.

C. Kit and Y. Wilks. 1999. Unsupervised learning of word boundary with description length gain. In *Proc. CoNLL99 ACL Workshop*, Bergen.

K. Koskenniemi. 1983. *Two-level morphology: A general computational model for word-form recognition and production*. Ph.D. thesis, University of Helsinki.

W. Li. 1992. Random texts exhibit Zipf's-Law-like word frequency distribution. *IEEE Transactions on Information Theory*, 38(6):1842–1845.

S. Neuvel and S. A. Fulop. 2002. Unsupervised learning of morphology without morphemes. In *Proc. ACL Workshop on Morphol. & Phonol. Learn.*, pp. 31–40.

J. Rissanen. 1989. *Stochastic Complexity in Statistical Inquiry*, vol. 15. World Scientific Series in Computer Science, Singapore.

P. Schone and D. Jurafsky. 2000. Knowledge-free induction of morphology using Latent Semantic Analysis. In *Proc. CoNLL-2000 & LLL-2000*, pp. 67–72.

M. G. Snover and M. R. Brent. 2001. A Bayesian model for morpheme and paradigm identification. In *Proc. 39th Annual Meeting of the ACL*, pp. 482–490.

M. G. Snover, G. E. Jarosz, and M. R. Brent. 2002. Unsupervised learning of morphology using a novel directed search algorithm: Taking the first step. In *Proc. ACL Worksh. Morphol. & Phonol. Learn.*, pp. 11–20.

H. Yu. 2000. Unsupervised word induction using MDL criterion. In *Proc. ISCSL*, Beijing.