Evaluating the Effect of Word Frequencies in a Probabilistic Generative Model of Morphology

Sami Virpioja and **Oskar Kohonen** and **Krista Lagus**
Aalto University School of Science
Adaptive Informatics Research Centre
P.O. Box 15400, FI-00076 AALTO, Finland
{oskar.kohonen,sami.virpioja,krista.lagus}@tkk.fi

Abstract

We consider generative probabilistic models for unsupervised learning of morphology. When training such a model, one has to decide what to include in the training data; e.g., should the frequencies of words affect the likelihood, and should words occurring only once be discarded. We show that for a certain type of models, the likelihood can be parameterized on a function of the word frequencies. Thorough experiments are carried out with Morfessor Baseline, evaluating the resulting quality of the morpheme analysis on English and Finnish test sets. Our results show that training on word types or with a logarithmic function of the word frequencies give similar scores, while a linear function, i.e., training on word tokens, is significantly worse.

1 Introduction

Unsupervised morphology learning is concerned with the task of learning models of word-internal structure. By definition, a probabilistic generative model describes the joint distribution of morphological analyses and word forms. An essential question is whether the morphological model represents *types*, that is, disregarding word frequencies in corpora, or *tokens*, i.e. fully appreciating the word frequencies. It has been observed that for the well-known Morfessor Baseline method (Creutz and Lagus, 2002; Creutz and Lagus, 2007), training on types leads to a large improvement in performance over tokens, when evaluating against a linguistic gold standard segmentation (Creutz and Lagus, 2004; Creutz and Lagus, 2005). A similar effect for a more recent method is reported by Poon et al. (2009). However, intuitively the corpus frequencies of words should be useful information for learning the morphology. In support of this intuition, behavioral studies regarding storage and processing of multi-morphemic word forms imply that the frequency of a word form plays a role in how it is stored in the brain: as a whole or as composed of its parts (Alegre and Gordon, 1999; Taft, 2004). In addition, the optimal morphological analysis may depend on the task to which the analysis is applied. In Morpho Challenge evaluations (Kurimo et al., 2010b), the winners of the different tasks are often different algorithms. For example, in machine translation, the reason might be that the frequent inflected word forms do not benefit from being split. However, it is not trivial to utilize token counts in generative models, since word tokens follow a power-law distribution (Zipf, 1932), and thus naive approaches will over-emphasize frequent word forms.

In this article, we consider whether the frequency information is inherently useful or not in unsupervised learning of morphology. We show that for a certain class of generative models, including those of the Morfessor methods, the word frequency acts as a weight in the likelihood function. We explicitly modify the distribution of words that the model approximates, also allowing choices *between* types and tokens.

Related to our approach, Goldwater et al. (2006) define a Bayesian two-level model where the first level generates word forms according to a multinomial distribution and the second level skews the distribution towards the observed power law distribution. For extreme parameter values of the second level process, the multinomial is trained with either types or tokens. For intermediate values, frequent words are emphasized, but not as much as when using token counts directly. They find experimentally that in morphological segmentation the best results are achieved when the parameter value is close to using only types, but emphasizes frequent words slightly. Their approach is elegant but computationally demanding. In contrast, our method is based on transforming the observed frequencies with a deterministic function, and therefore can be performed as a quick preprocessing step for existing algorithms.

Another intermediate option between types and tokens is given by Snyder and Barzilay (2008). Their morphological model generates bilingual phrases instead of words, and consequently, it is trained on aligned phrases that consist up to 4–6 words. The phrase frequencies are applied to discard phrases that occur less than five times, as they are likely to cause problems because of the noisy alignment. However, training is based on phrase types. Considering the frequencies of the words in this type of data, the common words will have more weight, but not as much as if direct corpus frequency was used.

We study the effect of the frequency information on the task of finding segmentations close to a linguistic gold standard. We use Morfessor Baseline, which is convenient due to its fast training algorithm. However, it has a property that causes it to arrive at fewer morphemes per words on average when the size of the training data grows (Creutz and Lagus, 2007). This phenomenon, which we refer to as undersegmentation, happens also when the model is trained on token counts rather than types, but it is not inherently related to the word frequency weighting in the class of models studied. Recently, Kohonen et al. (2010) showed how the amount of segmentation can be controlled by weighting the likelihood. In their semi-supervised setting, optimizing the weight improved the results considerably. This results in state of the art performance in Morpho Challenge 2010 (Kurimo et al., 2010a). In order to evaluate the effect of the frequency information without the problem of undersegmentation, we apply a similar likelihood weighting.

Another potential use for frequencies is noise reduction. Corpora often contain misspelled word forms and foreign names, but they are likely to occur very infrequently and are therefore removed if one discards rare word forms. It has been observed that pruning words that occur fewer times than a given threshold sometimes improves results in linguistic evaluations (Virpioja et al., 2010). We examine to what extent this improvement is explained by noise reduction, and to what extent it is explained by improving on undersegmentation.

2 Methods

In this section, we first consider generative probabilistic models in the task of learning morphology. We show that by making some simple assumptions, the data likelihood function can be parameterized on a function of the word frequencies. Then we describe the Morfessor Baseline model in this general framework.

2.1 Generative models of morphology

A generative model of morphology specifies the joint distribution $P(A = a, W = w \,|\, \boldsymbol{\theta})$ of words W and their morphological analyses A for given parameters $\boldsymbol{\theta}$.[1] W is an observed and A a hidden variable of the model. Here we assume that an analysis is a list of morpheme labels: $a = (m_1, \ldots, m_n)$. The probability of an analysis for a given word can be obtained by

$$\begin{aligned} & P(A = a \,|\, W = w, \boldsymbol{\theta}) \\ & = \frac{P(A = a, W = w \,|\, \boldsymbol{\theta})}{\sum_{\bar{a}} P(A = \bar{a}, W = w \,|\, \boldsymbol{\theta})}. \end{aligned} \tag{1}$$

Generative models can be trained with unlabeled data $\boldsymbol{D}$. For model classes with a large number of parameters, estimating the posterior probability of the parameters of a model, $P(\boldsymbol{\theta} \,|\, \boldsymbol{D}) \propto P(\boldsymbol{\theta}) P(\boldsymbol{D} \,|\, \boldsymbol{\theta})$, may be difficult. An alternative is to use a point estimate of the model parameters, $\boldsymbol{\theta}^*$, and apply that in Eq. 1. Instead of the simplest point estimate, *maximum likelihood* (ML), it is often better to apply *maximum a posteriori* (MAP), where a prior distribution is used to encode possible prior information about the model parameters:

$$\boldsymbol{\theta}^{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} \big\{ P(\boldsymbol{\theta}) \times P(\boldsymbol{D} \,|\, \boldsymbol{\theta}) \big\}. \tag{2}$$

Let $\boldsymbol{D}_W$ be a set of training data containing word forms. Assuming that the probabilities of the words are independent, the likelihood of the data can be calculated as

$$\begin{aligned} P(\boldsymbol{D}_W \,|\, \boldsymbol{\theta}) &= \prod_{j=1}^{|\boldsymbol{D}_W|} P(W = w_j \,|\, \boldsymbol{\theta}) \\ &= \prod_{j=1}^{|\boldsymbol{D}_W|} \sum_a P(A = a, W = w_j \,|\, \boldsymbol{\theta}). \end{aligned} \tag{3}$$

[1] We denote random variables with uppercase letters and their instances with lowercase letters.

Using the chain rule,

$$\begin{aligned} &P(A = a, W = w \mid \boldsymbol{\theta}) \\ &= P(A = a \mid \boldsymbol{\theta}) P(W = w \mid A = a, \boldsymbol{\theta}) \\ &= P(A = a \mid \boldsymbol{\theta}) \, \mathrm{I}(\mathrm{w}(a, \boldsymbol{\theta}) = w), \end{aligned} \tag{4}$$

where $\mathrm{w}(a, \boldsymbol{\theta})$ indicates the word form produced by the analysis a, and $\mathrm{I}(X) = 1$ if X is true and zero otherwise. Thus, the choice for $P(A = a \mid \boldsymbol{\theta})$ and $\mathrm{w}(a, \boldsymbol{\theta})$ defines the model class.

If we assume that the training data has $|\boldsymbol{D}_W|$ word types w_j with their respective counts c_j, the logarithm of the corpus likelihood is

$$\sum_{j=1}^{|\boldsymbol{D}_W|} c_j \ln \sum_a P(A = a, W = w_j \mid \boldsymbol{\theta}). \tag{5}$$

Using types or tokens for training the model can be seen as modifying the counts c_j with a function $f()$, where $f(c_j) = 1$ corresponds to training on types and $f(c_j) = c_j$ on tokens. Generally, this results in the *weighted log-likelihood*

$$\sum_{j=1}^{|\boldsymbol{D}_W|} f(c_j) \ln \sum_a P(A = a, W = w_j \mid \boldsymbol{\theta}), \tag{6}$$

where $f()$ maps the counts into non-negative values. In other words, if we assume that each instance of a word form is generated independently, the modified frequency $f(c_j)$ of that form becomes a proportional weight in the likelihood function. Thus, when training on tokens, the model aims to give higher probabilities to frequent word types compared to rare word types.

2.2 Morfessor Baseline

Morfessor Baseline (Creutz and Lagus, 2002; Creutz and Lagus, 2005; Creutz and Lagus, 2007) is a method for morphological segmentation: The analysis of a word is a list of its non-overlapping segments, morphs. The method is inspired by the Minimum Description Length (MDL) principle by Rissanen (1978) and tries to encode the words in the training data with a lexicon of morphs. It applies the two-part coding variant of MDL, which is equivalent to MAP estimation using a particular type of prior. The MDL derived priors prevent overfitting by assigning a low prior probability to models with a large number of parameters.

Following the notation by Kohonen et al. (2010), the model parameters $\boldsymbol{\theta}$ are:

- Morph type count, or the size of the morph lexicon, $\mu \in \mathbb{Z}_+$
- Morph token count, or the number of morphs tokens in the observed data, $\nu \in \mathbb{Z}_+$
- Morph strings $\sigma_1, \ldots, \sigma_\mu$, $\sigma_i \in \Sigma^*$
- Morph counts $(\tau_1, \ldots, \tau_\mu)$, $\tau_i \in \mathbb{Z}_+$, $\sum_i \tau_i = \nu$.

With non-informative priors, μ and ν can be neglected when optimizing. The morph string prior is based on the morph length distribution $P(L)$ and distribution $P(C)$ of characters over a character set Σ using the assumption that the characters are independent. For morph counts, the implicit non-informative prior $P(\tau_1, \ldots, \tau_\mu) = 1/\binom{\nu-1}{\mu-1}$ can be applied when μ and ν are known.

Each morph m_i in the lexicon has a probability of occurring in a word, $P(M = m_i \mid \boldsymbol{\theta})$, estimated from the count τ_i. A word is a sequence of morphs and the morph probabilities are assumed to be independent, so $\mathrm{w}(a, \boldsymbol{\theta}) = m_1 m_2 \ldots m_{|a|}$ and the probability of the analysis a is

$$P(A = a \mid \boldsymbol{\theta}) = \prod_{i=1}^{|a|} P(M = m_i \mid \boldsymbol{\theta}), \tag{7}$$

where m_i:s are the morphs in the analysis a.

The training algorithms of Morfessor apply the likelihood function only conditioned on the analyses of the observed words $\boldsymbol{A}$, $P(\boldsymbol{D}_W \mid \boldsymbol{A}, \boldsymbol{\theta})$. As before, an instance of $\boldsymbol{A}$ for the j:th word is a sequence of morphs: $a_j = (m_{j1}, \ldots, m_{j|a_j|})$. Furthermore, each word is assumed to have only a single analysis. For a known $\boldsymbol{a}$, the weighted log-likelihood (Eq. 6) is thus

$$\begin{aligned} &\ln P(\boldsymbol{D}_W \mid \boldsymbol{A} = \boldsymbol{a}, \boldsymbol{\theta}) \\ &= \sum_{j=1}^{|\boldsymbol{D}_W|} f(c_j) \sum_{i=1}^{|a_j|} \ln P(M = m_{ji} \mid \boldsymbol{\theta}), \end{aligned} \tag{8}$$

where m_{ij} is the i:th morph in word w_j. The number of morphs in the analysis, $|a_j|$, has a large effect on the probability of the word. Therefore the model prefers using a small number of morphs for words with a large $f(c_j)$.

The training algorithm of Morfessor Baseline minimizes the cost function $L(\boldsymbol{\theta}, \boldsymbol{a}, \boldsymbol{D}_W) = -\ln P(\boldsymbol{\theta}) - \ln P(\boldsymbol{D}_W \mid \boldsymbol{a}, \boldsymbol{\theta})$ by testing local changes to $\boldsymbol{a}$. The training algorithm is described, e.g., by Creutz and Lagus (2005). In the semi-supervised weighting scheme by Kohonen et al. (2010), the log-likelihood is weighted by a positive constant α, which is optimized for the chosen evaluation measure using a development set. After training, a Viterbi-like algorithm can be applied to find the optimal analysis for each word given the model parameters; a description of the procedure is provided, e.g., by Virpioja et al. (2010).

3 Experiments

The goal of our experiments is to find an optimal function $f()$ for the weighted log-likelihood in Eq. 6. We consider the following set of functions for the counts c_j:

$$f(x) = \begin{cases} 0 & \text{if } x < T \\ \alpha g(x) & \text{otherwise} \end{cases} \tag{9}$$

If $\alpha = 1$ and $g(x) = 1$ we train on word types (lexicon); if $\alpha = 1$ and $g(x) = x$, we train on tokens (corpus). In addition, we test a logarithmic function $g(x) = \ln(1 + x)$. The frequency threshold T can be used for pruning rare words from the training data. The global weight α modifies the balance between the likelihood and the model prior as in Kohonen et al. (2010). Both T, α and the function type $g(x)$ can be optimized for a given data set and a target measure. To ensure that we do not overlearn the data set for which we optimize the function parameters, we use a separate test set for the final evaluations.

We use Morfessor Baseline in the experiments, as it is fast enough for training a large number of models even with large training corpora. Our implementation was based on the Morfessor 1.0 software (Creutz and Lagus, 2005). The format of the input data is a list of words and their counts, so the function $f()$ is, in principle, trivial to apply as preprocessing. However, because the Morfessor prior assumes integer counts, the parameter α was implemented as a global weight for the likelihood. Otherwise, we modified the training data according to the respective function before training. The result of the logarithmic function was rounded to the nearest integer. We used the standard training algorithm and implicit morph length and frequency priors. For words not present in the training data, we applied the Viterbi algorithm to find the best segmentation, allowing new morphs with the approximate cost of adding them into the morph lexicon.

3.1 Data and evaluation

We used the English and Finnish data sets from Competition 1 of Morpho Challenge 2009 (Kurimo et al., 2010b). These languages were chosen because of their different morphological characteristics. Both sets were extracted from a three million sentence corpora. For English, there were $62,185,728$ word tokens and $384,903$ word types. For Finnish, there were $36,207,308$ tokens and $2,206,719$ types. The complexity of Finnish morphology is indicated by almost ten times larger number of word types than for English, while the numbers of word tokens are much closer.

We applied also the evaluation method of the Morpho Challenge competition.[2] The results of the morphological segmentation were compared to a linguistic gold standard analysis for a set of word types. Precision measures whether the word types that share morphemes in the proposed analysis have common morphemes also in the gold standard. Recall is calculated analogously by swapping the roles proposed and gold standard analyses. The final score is the F-measure, the harmonic mean of precision and recall.

Finnish gold standard was based on the morphological analyzer FINTWOL from Lingsoft, Inc., that applies the two-level model by Koskenniemi (1983). English gold standard was from the CELEX database. We applied the same final test sets as in Morpho Challenge, based on $10,000$ English word forms and $200,000$ Finnish word forms. For tuning the parameters of the weight function, we sampled a development set that did not contain any of the words in the final test set. The development set included $2,000$ word forms for English and $8,000$ word forms for Finnish.

3.2 Results

We trained Morfessor Baseline with the word frequencies set according to the three different function types. For each type, we optimized the cutoff parameter T and the weight parameter α by choosing values that gave the optimal F-measure on the development set. When $\alpha = 1.0$ and $T = 1$, the results correspond to those of the standard Morfessor Baseline. First, we varied only one parameter while the other one was fixed at one. Figure 1 shows the results for English and Figure 2

[2] Both the training data and evaluation scripts are available from the Morpho Challenge 2009 web page: `http://www.cis.hut.fi/morphochallenge2009/`

for Finnish using precision-recall curves. The best results are in the top-right corner. In solid lines, $\alpha = 1$ and T varied; in dashed lines, $T = 1$ and α varied. When precision is high and recall is low, the algorithm undersegments. With the constant function, either reducing α or increasing T improved recall at the expense of precision. In other words, pruning improves the results mostly by preventing undersegmentation, not by removing noise. With logarithmic or linear counts, increasing the frequency threshold did not improve recall, but there was no such problem with decreasing α. Especially for English, the linear counts did not provide as good results as the others.

Next, we optimized the F-measure for each function type by varying both T and α. Every parameter combination was not computed, but we concentrated on the areas where the locally optimal results were found. The results for English are presented in Tables 1–3 and the results for Finnish in Tables 4–6. If frequency information is useful for the model, we should see an improvement in results for the linear and logarithmic function over the constant one when the weight α and the threshold T are optimal. While the linear function performed worse than the others even with the optimal weighting, the logarithmic function provided small improvements over the constant function for both languages.

The optimal α was the largest for the constant function, smaller (English) or the same (Finnish) for the logarithmic function, and the smallest for the linear function. Smaller α means that the algorithm would undersegment more without the weight. The weights for Finnish were smaller than for English, which is explained by a larger number of word types in the training set. A possible reason for the same $\alpha = 0.01$ for Finnish when using constant and logarithmic functions is that the most of the likelihood cost is anyway due to the word forms observed only once, and the logarithmic function does not affect that.

Regarding the cutoff parameter T for English, the optimal frequency threshold was around 10–20 for constant and logarithmic functions, but only one for the linear function. A possible explanation is that rare words do not contain new morphological information, as they typically are uncommon nouns with no or only a single suffix. With the linear function, they get a very low weight in any case and cause no problems, but with the other functions, they are best to be excluded. For the Finnish data, the optimal frequency threshold was one for all three function types, so also the word forms occurring only once were useful for the algorithm. In an agglutinative language, such as Finnish, many valid inflected forms are very rare and therefore pruning does not remove only noise. While our results imply that it is better to use a smaller α than to prune, pruning infrequent words may still be useful in reducing computation time without sacrificing much accuracy.

Table 7 shows the results on the final test set. Again, using the word frequencies without optimized α and T clearly increase the problem of undersegmentation. In optimized cases, the results are more even. Note that unbalanced precision and recall imply that the tuning of the parameters did not completely succeed. For English, logarithmic counts gave higher F-measure also for the test set, but the difference to constant was not statistically significant according to the Wilcoxon signed-rank test. Linear counts gave clearly the worst results both for precision and recall. For Finnish, logarithmic counts did not give the improvement that the development set results promised: constant was slightly but significantly better. However, the slight undersegmentation indicates that it could be improved by fine tuning α. With linear counts, the F-measure was close, but still significantly lower.

Function	*Opt*	*Pre*	*Rec*	*F-m*
		English		
constant	no	76.13	48.97	**59.60**
logarithmic	no	87.76	31.77	46.65
linear	no	84.93	12.00	21.03
constant	yes	62.04	62.27	62.16
logarithmic	yes	57.85	67.62	**62.35**
linear	yes	53.96	56.42	55.16
		Finnish		
constant	no	89.50	15.70	**26.72**
logarithmic	no	91.24	11.95	21.13
linear	no	91.82	6.75	12.57
constant	yes	53.77	45.16	**49.09**
logarithmic	yes	57.87	42.06	48.72
linear	yes	48.86	47.37	48.10

Table 7: Precision (pre), recall (rec) and F-measure (F-m) on the final test set with the different function types for word frequencies. In optimized cases (opt), T and α are selected according to the best F-measure for the development set.

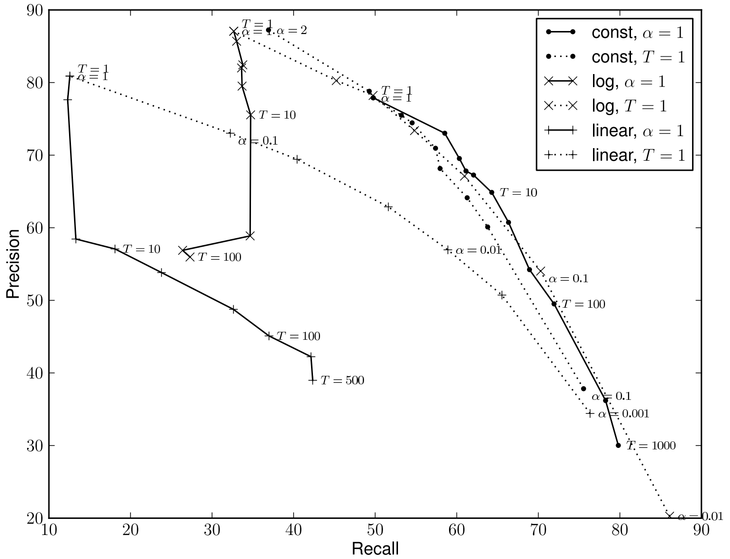

Figure 1: Precision-recall curves for English with constant (const), logarithmic (log), and linear frequency function types and varying function parameters α or T.

$T \setminus \alpha$	2	1.5	1.2	1.1	1	0.9	0.8	0.7	0.6
1	51.88	-	-	60.64	60.73	62.42	62.95	**63.46**	62.65
2	-	-	-	64.14	**64.97**	64.30	-	-	-
3	-	-	-	64.27	**64.60**	63.81	-	-	-
4	-	-	-	64.26	**64.30**	64.30	-	-	-
5	-	-	-	64.53	**64.55**	63.85	-	-	-
10	-	63.88	64.68	**65.30**	64.58	-	-	-	-
20	62.58	64.53	**65.29**	64.38	63.42	-	-	-	-
50	61.65	**63.13**	62.68	62.31	60.70	-	-	-	-

Table 1: Optimization results for English with $g(x) = 1$. Local optimum for each row (T) is written in boldface. The overall best results is underlined.

$T \setminus \alpha$	1	0.5	0.4	0.3	0.2	0.1
1	47.50	57.85	60.81	62.76	**63.88**	61.10
2	47.66	58.47	-	63.56	**63.80**	60.79
3	47.91	60.79	-	63.66	**64.12**	59.66
4	47.68	60.81	-	63.18	**64.53**	59.52
5	47.32	-	-	63.44	**64.47**	59.10
10	47.58	-	62.65	**64.87**	64.75	57.61
20	43.64	-	-	64.70	**65.57**	56.86
50	36.05	-	-	62.40	**63.93**	54.03

Table 2: Optimization results for English with $g(x) = \ln(1 + x)$.

$T \setminus \alpha$	1	0.1	0.05	0.02	0.01	0.005	0.001
1	21.75	44.73	51.10	56.69	**57.92**	57.19	47.45
2	21.24	-	-	56.31	**57.75**	57.07	-
5	21.67	-	-	55.98	**57.82**	57.03	-
10	27.48	-	-	56.11	**57.69**	56.40	-
20	33.01	-	-	55.67	57.50	-	-
50	39.09	-	-	-	57.10	-	-
100	40.64	-	-	-	-	-	-
200	42.19	-	-	-	-	-	-
500	40.60	-	-	-	-	-	-

Table 3: Optimization results for English with $g(x) = x$.

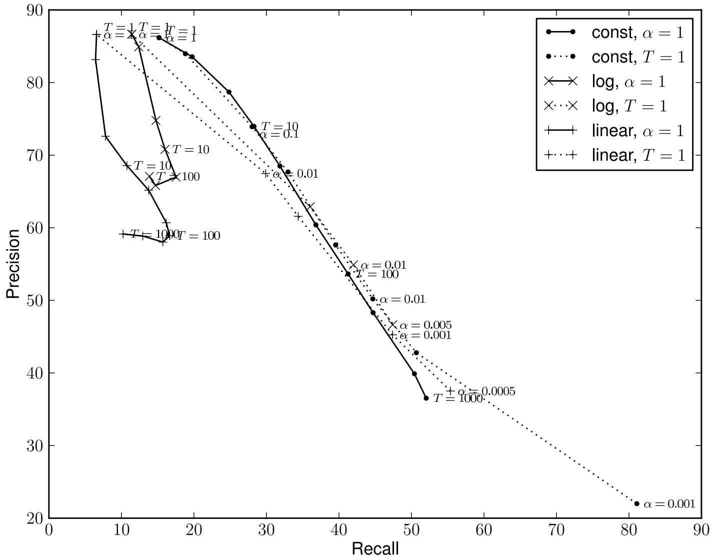

Figure 2: Precision-recall curves for Finnish with constant (const), logarithmic (log), and linear frequency function types and varying function parameters α or T.

$T \setminus \alpha$	1.2	1.1	1.0	0.5	0.2	0.1	0.05	0.02	0.01	0.005
1	-	-	25.83	30.74	-	40.64	44.35	46.91	**<u>47.27</u>**	46.40
2	-	-	31.96	36.86	-	45.07	**46.96**	46.80	45.35	-
5	-	-	37.75	41.89	46.15	**47.23**	46.83	-	39.10	-
10	-	-	40.92	44.37	**46.93**	46.61	-	-	33.85	-
20	-	-	43.49	46.15	**47.04**	45.88	-	-	-	-
50	-	-	45.75	**46.82**	45.70	42.17	-	-	-	-
100	-	46.29	**46.65**	46.50	-	-	-	-	-	-
200	46.19	**46.45**	46.43	-	-	-	-	-	-	-
500	44.69	**44.82**	44.54	-	-	-	-	-	-	-
1000	42.63	**43.17**	42.92	-	-	-	-	-	-	-

Table 4: Optimization results for Finnish with $g(x) = 1$. Local optimum for each row (T) is written in boldface. The overall best results is underlined.

$T \setminus \alpha$	1.0	0.5	0.2	0.05	0.02	0.01	0.005
1	20.20	-	-	-	45.81	**<u>47.60</u>**	47.02
2	21.62	29.81	36.74	44.43	46.96	**47.26**	46.28
5	24.66	-	-	46.03	**46.97**	46.28	-
10	26.11	-	-	-	46.70	-	-
20	27.80	-	-	-	-	-	-
50	24.03	-	-	-	-	-	-

Table 5: Optimization results for Finnish with $g(x) = \ln(1 + x)$.

$T \setminus \alpha$	1.0	0.1	0.01	0.005	0.001	0.0005
1	12.21	-	41.44	44.12	**<u>46.30</u>**	44.73
2	11.94	28.80	41.78	43.97	**45.94**	44.36
5	14.10	-	41.91	44.13	45.50	-
10	18.59	-	42.06	44.01	-	-
20	22.73	-	42.08	-	-	-
50	25.55	-	-	-	-	-
100	25.97	-	-	-	-	-
200	24.73	-	-	-	-	-
500	21.21	-	-	-	-	-

Table 6: Optimization results for Finnish with $g(x) = x$.

4 Conclusions

We showed that for probabilistic models, where word forms are generated independently, the word frequency acts as a relative weight in the likelihood function, changing how important the probabilities of the forms are to the likelihood. In the case of Morfessor Baseline, words with a large relative weight are segmented less, and vice versa. In the experiments, we trained Morfessor Baseline using three types of functions—constant, logarithmic, and linear—for the corpus frequencies of the words. Constant corresponds to learning on word types and linear on tokens, whereas logarithmic is between them. To overcome the model's tendency to undersegment, we used a likelihood weight optimized to give the best F-measure on a development set. While earlier results implied that learning on word types is the best option for this model when evaluated against linguistic gold standards, we showed that results of the same quality can also be obtained with logarithmic counts. In contrast, using corpus frequencies in a linear manner does not work as well. We also optimized a pruning threshold for the infrequent words. Pruning is simple and fast, but appears to work well only with the constant function type.

Acknowledgments

This work was funded by Graduate School of Language Technology in Finland and Academy of Finland.

References

Maria Alegre and Peter Gordon. 1999. Frequency effects and the representational status of regular inections. *Journal of Memory and Language*, 40:41–61.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30, Philadelphia, Pennsylvania, USA.

Mathias Creutz and Krista Lagus. 2004. Induction of a simple morphology for highly-inflecting languages. In *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology*, pages 43–51, Barcelona, July.

Mathias Creutz and Krista Lagus. 2005. Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0. Technical Report A81, Publications in Computer and Information Science, Helsinki University of Technology.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1), January.

Sharon Goldwater, Tom Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, pages 459–466. MIT Press, Cambridge, MA.

Oskar Kohonen, Sami Virpioja, and Krista Lagus. 2010. Semi-supervised learning of concatenative morphology. In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 78–86, Uppsala, Sweden, July.

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

Mikko Kurimo, Sami Virpioja, and Ville T. Turunen (Eds.). 2010a. Proceedings of the Morpho Challenge 2010 workshop. Technical Report TKK-ICS-R37, Aalto University School of Science and Technology, Department of Information and Computer Science, Espoo, Finland, September.

Mikko Kurimo, Sami Virpioja, Ville T. Turunen, Graeme W. Blackwood, and William Byrne. 2010b. Overview and results of Morpho Challenge 2009. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, volume 6241 of *Lecture Notes in Computer Science*, pages 578–597. Springer.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of NAACL HLT 2009*, pages 209–217.

Jorma Rissanen. 1978. Modeling by shortest data description. *Automatica*, 14:465–471.

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL-08: HLT*, pages 737–745, Columbus, Ohio, June.

Marcus Taft. 2004. Morphological decomposition and the reverse base frequency effect. *The Quarterly Journal of Experimental Psychology*, A 57:745–765.

Sami Virpioja, Oskar Kohonen, and Krista Lagus. 2010. Unsupervised morpheme analysis with Allomorfessor. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, volume 6241 of *Lecture Notes in Computer Science*, pages 609–616. Springer.

George Kingsley Zipf. 1932. *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, MA.