Morfessor 2.0: Toolkit for statistical morphological segmentation

Peter Smit[1]
peter.smit@aalto.fi

Sami Virpioja[2]
sami.virpioja@aalto.fi

Stig-Arne Grönroos[1]
stig-arne.gronroos@aalto.fi

Mikko Kurimo[1]
mikko.kurimo@aalto.fi

[1]Department of Signal Processing and Acoustics, Aalto University
[2]Department of Information and Computer Science, Aalto University

Abstract

Morfessor is a family of probabilistic machine learning methods for finding the morphological segmentation from raw text data. Recent developments include the development of semi-supervised methods for utilizing annotated data. Morfessor 2.0 is a rewrite of the original, widely-used Morfessor 1.0 software, with well documented command-line tools and library interface. It includes algorithmic improvements and new features such as semi-supervised learning, online training, and integrated evaluation code.

1 Introduction

In the morphological segmentation task, the goal is to segment words into morphemes, the smallest meaning-carrying units. Morfessor is a family of methods for unsupervised morphological segmentation. The first version of Morfessor, called Morfessor Baseline, was developed by Creutz and Lagus (2002) its software implementation, Morfessor 1.0, released by Creutz and Lagus (2005b). A number of Morfessor variants have been developed later, including Morfessor Categories-MAP (Creutz and Lagus, 2005a) and Allomorfessor (Virpioja et al., 2010). Even though these algorithms improve Morfessor Baseline in some areas, the Baseline version has stayed popular as a generally applicable morphological analyzer (Spiegler et al., 2008; Monson et al., 2010).

Over the past years, Morfessor has been used for a wide range of languages and applications. The applications include large vocabulary continuous speech recognition (e.g. Hirsimäki et al., 2006), machine translation (e.g. Virpioja et al., 2007), and speech retrieval (e.g. Arisoy et al., 2009). Morfessor is well-suited for languages with concatenative morphology, and the tested languages include Finnish and Estonian (Hirsimäki et al., 2009), German (El-Desoky Mousa et al., 2010), and Turkish (Arisoy et al., 2009).

Morfessor 2.0 is a new implementation of the Morfessor Baseline algorithm.[1] It has been written in a modular manner and released as an open source project with a permissive license to encourage extensions. This paper includes a summary of the Morfessor 2.0 software and a description of the demonstrations that will be held. An extensive description of the features in Morfessor 2.0, including experiments, is available in the report by Virpioja et al. (2013).

2 Morfessor model and algorithms

Models of the Morfessor family are generative probabilistic models that predict compounds and their analyses (segmentations) given the model parameters. We provide a brief overview of the methodology; Virpioja et al. (2013) should be referred to for the complete formulas and description of the model and its training algorithms.

Unlike older Morfessor implementations, Morfessor 2.0 is agnostic in regard to the actual data being segmented. In addition to morphological segmentation, it can handle, for example, sentence chunking. To reflect this we use the following generic terms: The smallest unit that can be split is an *atom* (letter). A *compound* (word) is a sequence of atoms. A *construction* (morph) is a sequence of atoms contained inside a compound.

2.1 Model and cost function

The cost function of Morfessor Baseline is derived using maximum a posteriori estimation. That is, the goal is to find the most likely parameters $\boldsymbol{\theta}$

[1] Morfessor 2.0 can be downloaded from the Morpho project website (http://www.cis.hut.fi/projects/morpho/) or GitHub repository (https://github.com/aalto-speech/morfessor).

given the observed training data $\boldsymbol{D}_W$:

$$\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}) p(\boldsymbol{D}_W \mid \boldsymbol{\theta}) \tag{1}$$

Thus we are maximizing the product of the model prior $p(\boldsymbol{\theta})$ and the data likelihood $p(\boldsymbol{D}_W \mid \boldsymbol{\theta})$. As usual, the cost function to minimize is set as the minus logarithm of the product:

$$L(\boldsymbol{\theta}, \boldsymbol{D}_W) = -\log p(\boldsymbol{\theta}) - \log p(\boldsymbol{D}_W \mid \boldsymbol{\theta}). \tag{2}$$

During training, the data likelihood is calculated using a hidden variable that contains the current chosen analyses. Secondly, it is assumed that the constructions in a compound occur independently. This simplifies the data likelihood to the product of all construction probabilities in the chosen analyses. Unlike previous versions, Morfessor 2.0 includes also the probabilities of the compound boundaries in the data likelihood.

For prior probability, Morfessor Baseline defines a distribution over the lexicon of the model. The prior assigns higher probability to lexicons that store fewer and shorter constructions. The lexicon prior consists of to parts, a product over the *form* probabilities and a product over the *usage* probabilities. The former includes the probability of a sequence of atoms and the latter the maximum likelihood estimates of the constructions. In contrast to Morfessor 1.0, Morfessor 2.0 currently supports only an implicit exponential length prior for the constructions.

2.2 Training and decoding algorithms

A Morfessor model can be trained in multiple ways. The standard batch training uses a local search utilizing recursive splitting. The model is initialized with the compounds and the full model cost is calculated. The data structures are designed in such way that the cost is efficient compute during the training.

In one epoch of the algorithm, all compounds in the training data are processed. For each compound, all possible two-part segmentations are tested. If one of the segmentations yields the lowest cost, it is selected and the segmentation is tried recursively on the resulting segments. In each step of the algorithm, the cost can only decrease or stay the same, thus guaranteeing convergence. The algorithm is stopped when the cost decreases less than a configurable threshold value in one epoch.

An extension of the Viterbi algorithm is used for decoding, that is, finding the optimal segmentations for new compound forms without changing the model parameters.

3 New features in Morfessor 2.0

3.1 Semi-supervised extensions

One important feature that has been implemented in Morfessor 2.0 are the semi-supervised extensions as introduced by Kohonen et al. (2010).

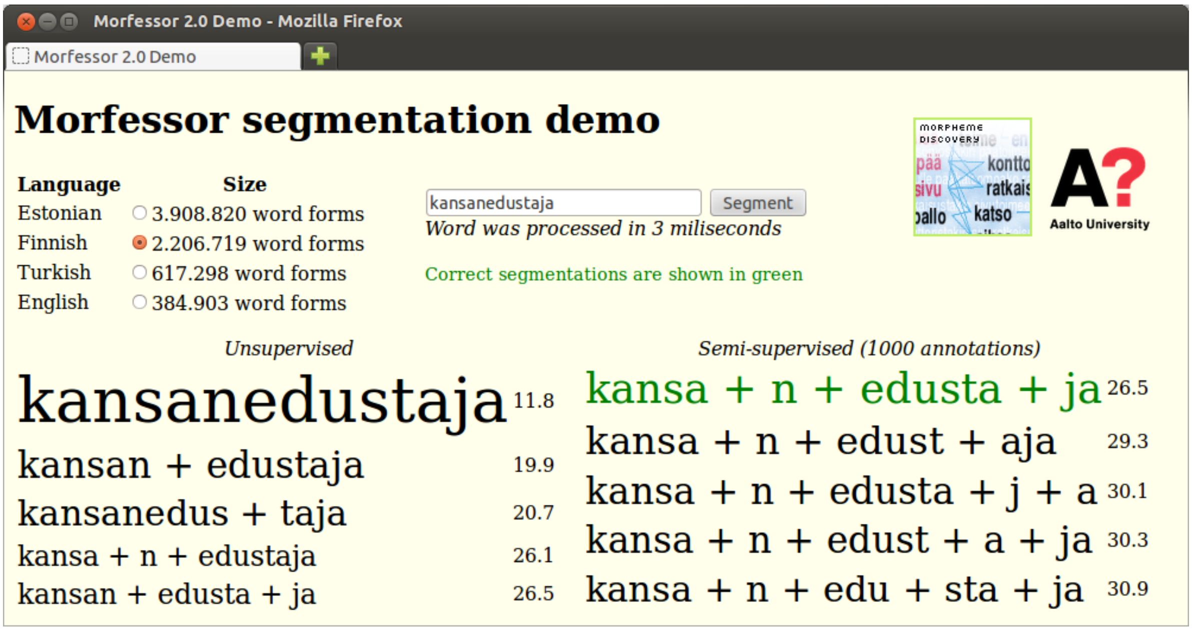
Morfessor Baseline tends to undersegment when the model is trained for morphological segmentation using a large corpus (Creutz and Lagus, 2005b). Oversegmentation or undersegmentation of the method are easy to control heuristically by including a weight parameter α for the likelihood in the cost function. A low α increases the priors influence, favoring small construction lexicons, while a high value increases the data likelihood influence, favoring longer constructions.

In semi-supervised Morfessor, the likelihood of an annotated data set is added to the cost function. As the amount of annotated data is typically much lower than the amount of unannotated data, its effect on the cost function may be very small compared to the likelihood of the unannotated data. To control the effect of the annotations, a separate weight parameter β can be included for the annotated data likelihood.

If separate development data set is available for automatic evaluation of the model, the likelihoods weights can be optimized to give the best output. This can be done by brute force using a grid search. However, Morfessor 2.0 implementation includes a simple heuristic for automatically tuning the value of α during the training, trying to balance precision and recall. A simple heuristic, which gives an equivalent contribution to the annotated data, is used for β.

3.2 On-line training

In addition to the batch training mode, Morfessor 2.0 supports on-line training mode, in which unannotated text is processed one compound at a time. This makes it simple to, for example, adapt pretrained models for new type of data. As frequent compounds are encountered many times in running text, an option is included for randomly skipping compounds and constructions that have been recently analyzed. The random skips can also be used to speed up the batch training.



Morfessor segmentation demo

Language	Size
Estonian	○ 3.908.820 word forms
Finnish	● 2.206.719 word forms
Turkish	○ 617.298 word forms
English	○ 384.903 word forms

kansanedustaja [Segment]

Word was processed in 3 miliseconds

Correct segmentations are shown in green

Unsupervised		*Semi-supervised (1000 annotations)*	
kansanedustaja	11.8	kansa + n + edusta + ja	26.5
kansan + edustaja	19.9	kansa + n + edust + aja	29.3
kansanedus + taja	20.7	kansa + n + edusta + j + a	30.1
kansa + n + edustaja	26.1	kansa + n + edust + a + ja	30.3
kansan + edusta + ja	26.5	kansa + n + edu + sta + ja	30.9

Figure 1: Screenshot from the Morfessor 2.0 demo.

3.3 Integrated evaluation code

One common method for evaluating the performance of a Morfessor model is to compare it against a gold standard segmentation using segmentation boundary precision and recall (BPR). To make the evaluation easy, the necessary tools for calculating a variant of the BPR metric by Virpioja et al. (2011) are included in Morfessor 2.0. For significance testing when comparing multiple models, we have included the Wilcoxon signed-rank test. Both the evaluation code and statistical testing code are accessible from both the command line and the library interface.

3.4 N-best segmentation

In order to generate multiple segmentations for a single compound, Morfessor 2.0 includes a n-best Viterbi algorithm. It allows extraction of all possible segmentations for a compound and the probabilities of the segmentations.

4 Demonstration

In the demonstration session, multiple features and usages of Morfessor will be shown.

4.1 Web-based demonstration

A live demonstration will be given of segmenting text with Morfessor 2.0 for different language and training data options. In a web interface, the user can choose a language, select the size of the training corpus and other options. After that a word can be given which will be segmented using n-best Viterbi, showing the 5 best results.

A list of planned languages can be found in Table 1. A screen shot of the demo interface is shown in Figure 1.

Languages	# Words	# Word forms
English	62M	384.903
Estonian	212M	3.908.820
Finnish	36M	2.206.719
German	46M	1.266.159
Swedish	1M	92237
Turkish	12M	617.298

Table 1: List of available languages for Morfessor 2.0 demonstration.

4.2 Command line interface

The new command line interface will be demonstrated to train and evaluate Morfessor models from texts in different languages. A diagram of the tools is shown in Figure 2

4.3 Library interface

Interfacing with the Morfessor 2.0 Python library will be demonstrated for building own scientific experiments, as well as integrating Morfessor in bigger project. Also the code of the Web based demonstration will be shown as an example.

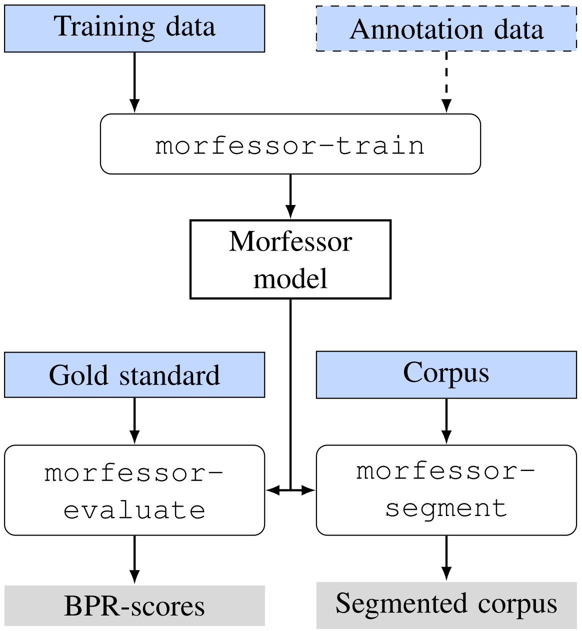


Figure 2: The standard workflow for Morfessor command line tools

Acknowledgements

The authors have received funding from the EC's 7th Framework Programme (FP7/2007–2013) under grant agreement n°287678 and the Academy of Finland under the Finnish Centre of Excellence Program 2012–2017 (grant n°251170) and the LASTU Programme (grants n°256887 and 259934). The experiments were performed using computer resources within the Aalto University School of Science "Science-IT" project.

References

E. Arisoy, D. Can, S. Parlak, H. Sak, and M. Saraclar. 2009. Turkish broadcast news transcription and retrieval. *Audio, Speech, and Language Processing, IEEE Transactions on*, 17(5):874–883.

M. Creutz and K. Lagus. 2002. Unsupervised discovery of morphemes. In Mike Maxwell, editor, *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30. Association for Computational Linguistics, July.

M. Creutz and K. Lagus. 2005a. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of AKRR'05, International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 106–113, Espoo, Finland, June. Helsinki University of Technology.

M. Creutz and K. Lagus. 2005b. 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.

A. El-Desoky Mousa, M. Ali Basha Shaik, R. Schluter, and H. Ney. 2010. Sub-lexical language models for German LVCSR. In *Spoken Language Technology Workshop (SLT), 2010 IEEE*, pages 171–176. IEEE.

T. Hirsimäki, M. Creutz, V. Siivola, M. Kurimo, S. Virpioja, and J. Pylkkönen. 2006. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech & Language*, 20(4):515–541.

T. Hirsimäki, J. Pylkkönen, and M. Kurimo. 2009. Importance of high-order n-gram models in morph-based speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 17(4):724–732.

O. Kohonen, S. Virpioja, and K. 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. Association for Computational Linguistics.

C. Monson, K. Hollingshead, and B. Roark. 2010. Simulating morphological analyzers with stochastic taggers for confidence estimation. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, pages 649–657. Springer.

S. Spiegler, B. Golénia, K. Shalonova, P. Flach, and R. Tucker. 2008. Learning the morphology of zulu with different degrees of supervision. In *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*, pages 9–12. IEEE.

S. Virpioja, J. Väyrynen, M. Creutz, and M. Sadeniemi. 2007. Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner. In *Proceedings of the Machine Translation Summit XI*, pages 491–498, Copenhagen, Denmark, September.

S. Virpioja, O. Kohonen, and K. Lagus. 2010. Unsupervised morpheme analysis with Allomorfessor. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, volume 6241 of *LNCS*, pages 609–616. Springer Berlin / Heidelberg.

S. Virpioja, V. Turunen, S. Spiegler, O. Kohonen, and M. Kurimo. 2011. Empirical comparison of evaluation methods for unsupervised learning of morphology. *TAL*, 52(2):45–90.

S. Virpioja, P. Smit, S. Grönroos, and M. Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for Morfessor Baseline. Report 25/2013 in Aalto University publication series SCIENCE + TECHNOLOGY, Aalto University, Finland.