Morfessor and VariKN machine learning tools for speech and language technology

Vesa Siivola, Mathias Creutz and Mikko Kurimo

Adaptive Informatics Research Centre, Helsinki University of Technology, Espoo, Finland

Firstname.Lastname@tkk.fi

10.21437/Interspeech.2007-446

Abstract

This paper introduces two recent open source software packages developed for unsupervised natural language modeling. The Morfessor program segments words automatically into morpheme-like units without any rule-based morphological analyzers. The VariKN toolkit trains language models producing a compact set of high-order n-grams utilizing state-of-art Kneser-Ney smoothing. As an example, this paper shows how to construct a language model for speech recognition in multiple languages utilizing only a minimal amount of linguistic resources. Morfessor and VariKN also have other applications in text understanding, information retrieval and machine translation. Unsupervised machine learning techniques are particularly well suited for the development of systems for less-resourced languages, because they do not depend on manually designed morphological or syntactical analyzers or annotated data.

Index Terms: variable length language modeling, speech recognition, unsupervised morphology, open source software

1. Introduction

The rapid growth of digital media has brought about large amounts of data written and spoken in languages for which there are few language processing tools available. This has encouraged people to direct funding for the development of machine translation, information retrieval and speech recognition for many new languages.

Porting existing tools to new languages is not straightforward. Many current language technology solutions are building on large hand-crafted lexica, pronunciation dictionaries, tree banks and morphological analyzers. Even if the tools themselves could, in principle, work in the new target languages, the available resources would not be sufficient for the creation of the required building blocks. In practice, it can turn out that the whole language modeling approach, and thus, the design principles of the tools, will need to be revised. This is the case, for example, in moving from English to highly inflected or agglutinative languages, Finnish and Estonian.

Instead of directing new resources into the manual construction of building blocks for traditional language technology tools, it is often better to develop new tools that utilize machine learning methods to automatically derive models from data. Especially for less-resourced and morphologically rich languages, algorithms capable of unsupervised learning are most useful, because they can utilize poorly annotated and formatted training data that is becoming available.

The Morfessor and VariKN program packages both implement unsupervised machine learning methods to obtain language models that emerge when maximizing corpus coverage while minimizing the model size. The algorithms are language independent in such a way that any suitably preprocessed large text corpus can be used as training data. The corpus may even include words and sentences in several different languages. Both packages have first been utilized for unlimited vocabulary speech recognition at Helsinki University of Technology (TKK) [1, 2]. As the main results, it is shown that the tools can successfully produce effective language models for different languages without the need for expert knowledge of the target language. In speech recognition, the language models have been successfully used to cover a practically unlimited vocabulary in three agglutinative languages: Finnish, Turkish and Estonian [2]. Other remarkable results are the online speech recognition demos and PASCAL Morpho Challenge competitions [3].

2. Morfessor

Morfessor is an unsupervised data-driven method for the segmentation of words into morpheme-like units. The general idea is to discover as compact a description of the input text corpus as possible. Substrings occurring frequently enough in several different word forms are proposed as *morphs*, and the words in the corpus are then represented as a concatenation of morphs, e.g., 'hand, hand+s, left+hand+ed, hand+ful'. Through maximum a posteriori optimization (MAP), an optimal balance is sought between the compactness of the inventory of morphs, i.e., the *morph lexicon*, versus the compactness of the representation of the corpus.

It has been shown (e.g.,[4, 5, 6, 7, 8]) that models based on the above approach produce segmentations that resemble linguistic morpheme segmentations, when formulated mathematically in a probabilistic framework or equivalently using the Minimum Description Length (MDL) principle [9]. Similarly, a hierarchical Dirichlet model has been used in combination with morph bigram probabilities [10].

The Morfessor model has been developed over the years, and different model versions exist. Two versions are discussed in the current paper: the oldest and simplest so-called *Morfessor Baseline* algorithm, as well as the most recent and most advanced so-called *Morfessor Categories-MAP* algorithm.

2.1. Morfessor Baseline

The Morfessor Baseline algorithm was originally introduced in [7], where it was called the "Recursive MDL" method. Additionally, the Baseline algorithm is described in [1]. The Baseline method is a *context-independent* splitting algorithm. This means that *morphotactic violations* may occur; for instance, the English segmentations 'wing+s' and 's+wing' are equally good, according to the model. Additionally, *undersegmentation* of frequent strings and *oversegmentation* of rare strings are fairly common errors, because the most concise representation is ob-

tained when any frequent string is stored as a whole in the lexicon (e.g., English 'having, soldiers, states, seemed'), whereas infrequent strings are better coded in parts (e.g., 'or+p+han, s+ed+it+ious, vol+can+o').

Morfessor Baseline can be used alone or as a baseline for bootstrapping the *context-dependent* Categories-MAP model version.

2.2. Morfessor Categories-MAP

Morfessor Categories-MAP makes use of morph categories; the segmentation of the corpus is modeled using a Hidden Markov model (HMM) with transition probabilities between categories and emission probabilities of morphs from categories. Three categories are used: *prefix, stem*, and *suffix* and an additional *non-morpheme* (or *noise*) category. More details can be found in [11].

The more advanced structure of Categories-MAP typically enables this model version to obtain more accurate morpheme segmentations than the Baseline method (when evaluated against a linguistic gold standard). For instance, the English morph '-s' is identified as a good suffix candidate, which can occur in the word 'wing+s', but it is not proposed word-initially in 'swing'.

2.3. Grapheme-to-phoneme mapping

In many languages (e.g., Finnish, Estonian, Turkish), the spelling of a word indicates the pronunciation of the word. More or less, there is a one-to-one correspondence between letters (graphemes) and phonemes. When splitting the word into parts, the pronunciation of the parts in isolation does not differ much from the pronunciation of the parts in context.

To cope with less obvious grapheme-to-phoneme mappings (e.g., in languages such as English and French), maximum likelihood alignment can be performed. In the first stage, spelled words are split using Morfessor. Next, segmentations of the pronunciations of the words are obtained using maximum-likelihood alignment of the characters in both strings (spelling vs. pronunciation). Breaks are inserted into the pronunciation at the locations given by the alignment.

In cases where one spelled morph gets different pronunciations in different contexts the different variants are made unique through numbering. The language model can then learn which variant to use in which context. For example, in English this would correspond to having two versions of the morph 'hid': 'hid1' (pronounced [hId]) and 'hid2' (pronounced [haId]). These morphs can be used, e.g., in the word forms 'hid' vs. 'hiding': 'hid1' vs. 'hid2 + ing'.

3. VariKN language modeling toolkit

Kneser-Ney smoothing [12] has been shown to be an excellent smoothing method for n-gram models [13, 14]. The VariKN language modeling toolkit is a specialized toolkit for building, pruning and growing Kneser-Ney smoothed models.

For most smoothing methods, the probability distributions of n-grams of one order do not depend on the existence of the probability distributions of other orders. In Kneser-Ney smoothing, a lower order probability distribution is modified to take into account what is modeled by the higher order probability distributions.

3.1. About variable order language models

A simple way of choosing, which n-grams to include in the language model is to determine the length of the n-grams to be modeled and include all n-grams up to that order that were seen in the training set. This is called a full model in this paper. Full models are inefficient, since some n-grams affect the overall probability distribution only insignificantly while some other n-grams are likely to never be used. The n-grams that do not affect the overall probability distribution much can be removed by pruning: if removing the n-gram does not change the likelihood of the training data significantly, the n-gram can be removed. Count cutoffs, on the other hand, remove the n-grams that have been seen fewer times than some set threshold. The motivation is that the infrequently seen n-grams are not likely to be seen again and the probability estimates are not likely to be accurate since only little data was found for estimating the parameters.

Variable order n-gram models are a good match for subword based language modeling. The modeling context can be extended over several shorter morphs if necessary. On the other hand, if a long context does not help in predicting a morph, memory can be saved by only using a few low order n-grams.

3.2. Likelihood pruning and count cutoffs

Several methods exist for pruning n-gram models [16, 17, 18]. Most of the methods do not take into account that with Kneser-Ney smoothing, the probability distributions of different orders depend on each other. Kneser pruning does take into account this fact, but there are other approximations that degrade the performance of the algorithm. The VariKN toolkit implements a pruning algorithm, which has been shown to give significantly better results for Kneser-Ney smoothed models than the other smoothing methods [19].

Often, count cutoffs are implemented so that the n-grams seen less frequently than a threshold are removed before the model is estimated. Additionally, the toolkit implements another variant: When n-grams are removed the lower order distributions are modified accordingly. Good results are often achieved by combining likelihood pruning and count cutoffs.

3.3. Growing

Generally, the starting point of the pruning algorithms is a full model, possibly with count cutoffs. However, if a high order n-gram model is desired, it is often impossible to construct such a model due to high memory consumption. Growing algorithms start from a small model and use a greedy search to find the most useful n-grams [20, 21]. They can generate high order models, since only some n-grams from each model order are included. The VariKN toolkit implements an growing algorithm, which takes into account the properties of Kneser-Ney smoothing and can provide an excellent starting point for the pruning algorithms [19].

4. Putting it all together

Figure 1 depicts the whole process of generating a VariKN language model based on morphs learned in an unsupervised way using Morfessor:

1. **Morfessor.** A morph segmentation model is trained using Morfessor. Both the Morfessor Baseline and Categories-MAP model versions are publicly available under GNU GPL (General Public License) at `http://www.cis.hut.fi/projects/morpho/`.

Morfessor takes as input a list containing all word forms occurring in the text corpus together with the numbers of occurrences of the words. In practice, if the Morfessor Baseline version is used, morph segmentations that are closer to linguistic morpheme segmentations are usually obtained if each word frequency is set to 1 rather than its real value. This reduces the dominance of frequent word forms in the model.
If the Morfessor Categories-MAP version is used, word frequencies need not be altered. One model parameter (the so-called perplexity threshold) must be set to an appropriate value. The value depends on the size of the corpus and the morphological structure of the language. Try a value between 10 and 50 to begin with.

2. **Viterbi segmentation.** It is possible to train a segmentation model on some set of words and then use this model to segment a larger set of words using the Viterbi algorithm (included in the Morfessor software packages). This may be necessary if the Morfessor Baseline algorithm is used and the resulting morph inventory is too large (for some intended purpose). Since the morph inventory discovered by the Morfessor Baseline algorithm is larger the more training data there is, the training set can be reduced by filtering out the least frequent word forms. The rarest words are excluded from the model training, but nonetheless segmentations for these words can be obtained by using the Viterbi algorithm to select the most likely segmentation according to the (smaller) model.

3. **Pronunciation of the morphs.** If there is no direct grapheme-to-phoneme mapping in the language studied, maximum-likelihood alignment can be performed in order to construct a pronunciation lexicon of morphs (see Sec. 2.3). As input, a standard pronunciation lexicon of words is necessary for at least part of the words. The necessary program will be available at `http://www.cis.hut.fi/projects/morpho/`.

4. **Modeling of word and sentence boundaries.** In word-based n-gram models, each lexicon entry is implicitly assumed to end in a word break. No such assumption can be made for a lexicon based on sub-word units. One way of modeling word breaks is to mark word-final morphemes using a special symbol, e.g., `word# break s# are# easi ly# model ed#`. Another (usually preferable) way is to add a special word break token between each split word, e.g., `word <w> break s <w> are <w> easi ly <w> model ed <w>`. The language model can be estimated as usual and is also capable of predicting where the word breaks should be placed.
Sentence breaks are often modeled similarly; special tokens for sentence start `<s>` and end `</s>` are added into the training corpus. It is customary not to use the contexts that cross a sentence boundaries in the n-gram models.

5. **Estimation of an n-gram language model.** The VariKN language modeling toolkit is available under GNU LGPL (Lesser General Public License) at `http://varikn.forge.pascal-network.org/`. For the estimation of an n-gram model, suitable growing and pruning parameters need to be determined. It is usually best to grow as large model as possible for the pruning algorithm. For producing a relatively small model, we suggest using 0.1 for growing and 0.25 for pruning. The training set needs to be partitioned into a set, where the n-gram probabilities are trained on and a held-out set, on which the language model discount parameters are optimized. If there is a sufficient amount of training data left in the main set, a suitable size for the held-out set is around 100 000 tokens.

6. **Evaluation of the language model.** Language model performance is often measured by calculating how well the probability distribution of the model fits test data. The result is usually normalized by the number of words in the test set; also the scores of subword based models should be normalized by the number of words for meaningful results. The VariKN toolkit reports both the cross-entropy and perplexity of the model relative to a test set.

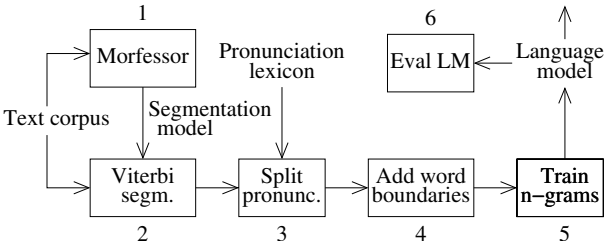

Figure 1: *Putting it all together: how to train segmentation model using the Morfessor Baseline algorithm, and how to further train an n-gram model based on morphs.*

5. Performance

A few experimental results illustrate the benefits of using the proposed tools.

Table 1 shows how accurately Morfessor succeeds in the placement of morpheme boundaries on four data sets of different languages (English, Finnish, Turkish, and the dialect of Arabic spoken in Egypt). Typically, the Categories-MAP model version outperforms the Baseline version, but for the "morphologically poor" language English, the difference is rather small.

Table 1: *Morpheme segmentation accuracy [%] obtained by Morfessor in comparison to a linguistic gold standard. The figures are F-measures (harmonic mean of precision and recall) of the discovery of morpheme boundaries. The sizes of the data sets are also shown (token and type counts).*

	English	Finnish	Turkish	EgyptArabic
Baseline	66.0	54.2	51.3	41.7
Categories-MAP	66.2	66.4	70.7	68.1
Word tokens	24M	32M	17M	150k
Word types	170k	1.6M	580k	17k

Table 2 summarizes speech recognition experiments on four languages: the Finnish and Turkish results have earlier been reported in [3], and the Estonian results in [22]. The figures show that morph-based recognition outperforms standard word-based recognition, except for Egyptian Arabic. The Arabic data set is very small, and the number of different word forms is small, which reduces the benefits of morph-based modeling. Additionally, the Arabic "templatic" morphology with non-contiguous morphemes poses special difficulties. It is pleasant to see that the morphs obtained in an unsupervised manner from unannotated text lose very little (if any) to morphs obtained from manually designed morphological analyzers. (It can further be noted that the more advanced Morfessor Categories-MAP version does not outperform the simpler Baseline method; all *morph* figures in Table 2 correspond to the Baseline method.)

Figure 2 shows the n-gram distribution by n-gram order; a full 5-gram model and a grown model are compared. The models were trained from a Finnish corpus of 150 million words (460 million morphs) and include approximately 200 million n-

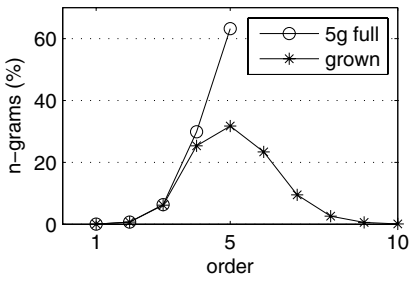

Figure 2: *N-gram distribution by n-gram order for Finnish; a full 5-gram model and a grown model.*

grams each. The higher order n-grams are useful because the grown model obtains lower perplexity on the test set than the full 5-gram model; 7600 for the grown model and 8600 for the full model. Note that the perplexity of some languages is naturally much higher than English due to the morphology of the language. If the perplexity is normalized by the number of sentences instead of the number of words, the values for Finnish and English are similar [19].

Table 2: *Automatic speech recognition performance: word error rates are reported for three alternative approaches: standard* word *model,* morph*s obtained using the Morfessor Baseline algorithm, and* grammatical *morphs obtained using manually designed morphological analyzers.*

	Finnish	Estonian	Turkish	EgyptArabic
word	17.9	53.1	32.6	57.7
morph	9.8	39.4	31.4	58.8
grammatical	9.6	38.7	31.4	59.1

6. Conclusion

Descriptions and instructions for using two recent open source software packages developed at Helsinki University of Technology are presented in this paper. Unsupervised segmentation of words into morpheme-like units is performed by Morfessor and variable length n-gram language models are built using the VariKN toolkit. The tools are language independent and easy to use. This paper shows how they can be used to construct language models for large vocabulary speech recognition in multiple languages, where a traditional word-based language model performs poorly. The tools also have other applications in text understanding, information retrieval and machine translation. Unsupervised machine learning techniques like these are particularly well suited for the development of systems for less-resourced languages, because they can operate on raw text data and do not depend on manually designed morphological or syntactical analyzers or annotated data.

7. References

[1] T. Hirsimäki, M. Creutz, V. Siivola, M. Kurimo, S. Virpioja, and J. Pylkkönen, "Unlimited vocabulary speech recognition with morph language models applied to Finnish," *Computer Speech and Language*, vol. 20, no. 4, pp. 515–541, 2006.

[2] M. Kurimo, A. Puurula, E. Arisoy, V. Siivola, T. Hirsimäki, J. Pylkkönen, T. Alumäe, and M. Saraclar, "Unlimited vocabulary speech recognition for agglutinative languages," in *Proc. HLT-NAACL*, pp. 487-494, 2006.

[3] M. Kurimo, M. Creutz, M. Varjokallio, E. Arisoy, and M. Saraclar, "Unsupervised segmentation of words into morphemes – Morpho Challenge 2005: Application to automatic speech recognition," in *Proc. ICSLP*, pp. 1021-1024, 2006.

[4] C. G. de Marcken, "Unsupervised language acquisition," Ph.D. dissertation, MIT, 1996.

[5] M. R. Brent, "An efficient, probabilistically sound algorithm for segmentation and word discovery," *Machine Learning*, vol. 34, pp. 71–105, 1999.

[6] J. Goldsmith, "Unsupervised learning of the morphology of a natural language," *Computational Linguistics*, vol. 27, no. 2, pp. 153–198, 2001.

[7] M. Creutz and K. Lagus, "Unsupervised discovery of morphemes," in *Proc. SIGPHON/ACL*, 2002, pp. 21–30.

[8] M. Creutz, "Induction of the morphology of natural language: Unsupervised morpheme segmentation with application to automatic speech recognition," Ph.D. dissertation, Helsinki University of Technology, 2006.

[9] J. Rissanen, *Stochastic complexity in statistical inquiry theory*. World Scientific Publishing Co., Inc., 1989.

[10] S. Goldwater, T. L. Griffiths, and M. Johnson, "Contextual dependencies in unsupervised word segmentation," in *Proc. Coling/ACL*, 2006, pp. 673–680.

[11] M. Creutz and K. Lagus, "Inducing the morphological lexicon of a natural language from unannotated text," in *Proc. AKRR*, 2005, pp. 106–113.

[12] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Proc. ICASSP*, 1995, pp. 181–184.

[13] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech and Language*, vol. 13, no. 4, pp. 359–393, 1999.

[14] J. Goodman, "Exponential priors for maximum entropy models," in *Proc. HLT-NAACL*, 2004, pp. 305–312.

[15] J. Goodman, "A bit of progress in language modeling, extended version," Microsoft Research, Tech. Rep. MSR-TR-2001-72, 2001.

[16] K. Seymore and R. Rosenfeld, "Scalable backoff language models," in *Proc. ICSLP*, 1996, pp. 232–235.

[17] A. Stolcke, "Entropy-based pruning of backoff language models," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, 1998, pp. 270–274.

[18] R. Kneser, "Statistical language modeling using a variable context length," in *Proc. ICSLP*, 1996, pp. 494–497.

[19] V. Siivola, T. Hirsimäki, and S. Virpioja, "On growing and pruning Kneser-Ney smoothed n-gram models," *IEEE Trans. Speech Audio Language Process.*, 2007, accepted for publication.

[20] E. S. Ristad and R. G. Thomas, "New techniques for context modeling," in *Proc. ACL*, 1995, pp. 220–227.

[21] T. R. Niesler and P. C. Woodland, "Variable-length category n-gram language models," *Computer Speech and Language*, vol. 13, no. 1, pp. 99–124, 1999.

[22] A. Puurula and M. Kurimo, "Vocabulary decomposition for Estonian open vocabulary speech recognition," in *Proc. ACL*, 2007, accepted for publication.