Unsupervised Word Decomposition with the Promodes Algorithm

Sebastian Spiegler, Bruno Golénia, Peter Flach

Machine Learning Group, Computer Science Department, University of Bristol, UK,
{spiegler, goleniab, flach}@cs.bris.ac.uk

Abstract. We present Promodes, an algorithm for unsupervised word decomposition, which is based on a probabilistic generative model. The model considers segment boundaries as hidden variables and includes probabilities for letter transitions within segments. For the Morpho Challenge 2009, we demonstrate three versions of Promodes. The first one uses a simple segmentation algorithm on a subset of the data and applies *maximum likelihood estimates* for model parameters when decomposing words of the original language data. The second version estimates its parameters through *expectation maximization* (EM). A third method is a *committee of unsupervised learners* where learners correspond to different EM initializations. The solution is found by majority vote which decides whether to segment at a word position or not. In this paper, we describe the probabilistic model, parameter estimation and how the most likely decomposition of an input word is found. We have tested Promodes on non-vowelized and vowelized Arabic as well as on English, Finnish, German and Turkish. All three methods achieved competitive results.

1 Introduction

Morphological analysis can be defined as study of the internal word structure [1]. According to [2], there are four tasks involved in morphological analysis: 1) decomposing words into morphemes, 2) building a morpheme dictionary, 3) defining morphosyntactical rules stating how morphemes are combined to valid words and 4) defining morphophonological rules specifying phonological changes when combining morphemes. For the Morpho Challenge, the task is *unsupervised morpheme analysis* of words contained in a word list using a generic algorithm without any further information.

Related Work Goldsmith [3] presented the morphological analyzer *Linguistica* which learns *signatures*. A similar approach has been chosen by Monson [4] who developed *Paramor*, an algorithm which induces *paradigmatic structures* of morphology. *Morfessor* is a model family for unsupervised morphology induction developed by Creutz et al. [5]. The two main members of this family are *Morfessor baseline* based on minimum description length (MDL) and *Morfessor Categories-MAP* with a probabilistic maximum a posteriori (MAP) framework and mor-

pheme categories.[1] Linguistica, Paramor and Morfessor carry out morphological analysis in terms of word decomposition, learning a morpheme dictionary and finding morphosyntactical rules. Other approaches [6, 7] focused on word decomposition by analyzing words based on *transition probabilities* or *letter successor variety* which originates in Harris' approach [8]. Moreover, Snover [9] described a *generative model* for unsupervised learning of morphology, however, it differs from ours. Snover searched, similar to Monson, for paradigms, whereas we are interested in word decomposition based on the probability of having a boundary in a certain position and the resulting letter transition of morphemes. The remainder of the paper is structured as follows. In Sec. 2 we present PROMODES, its mathematical model, the parameter estimation and word decomposition. In Sec. 3 and 4 experiments are explained, results analysed and conclusions drawn.

2 Algorithm

The PROMODES algorithm is based on a probabilistic generative model which can be used for word decomposition when fully parameterized. Its parameters can be estimated using *expectation maximization (EM)* [10] or by computing *maximum likelihood estimates (MLE)* from a pre-segmented training set. Independently of the parameter estimation, either a single model is used for decomposition or a set of separate models as a *committee of unsupervised learners* where η different results are combined by majority vote. In Sec. 2.1 we will introduce the PGM and show how we apply it to find a word's best segmentation. Subsequently, in Sec. 2.2 we will explain how we estimate model parameters and in Sec. 2.3 we demonstrate how a committee of unsupervised learners is used to decompose words.

2.1 Probabilistic Generative Model

A *probabilistic generative model (PGM)* is used to describe the process of data generation based on observed variables X and target variables Y with the goal of forming a conditional probability distribution $Pr(Y|X)$. In morphological analysis the observables correspond to the original words and the hidden variables to their segmentations. A word w_j from a list W with $1 \leq j \leq |W|$ consists of n letters and has $m = n-1$ positions for inserting boundaries. A word's segmentation b_j is described by a binary vector $(b_{j1}, \ldots, b_{jm})$. A boundary value b_{ji} is $\{0, 1\}$ depending on whether a boundary is inserted or not in i with $1 \leq i \leq m$. A letter transition t_{ji} consists of letter $l_{j,i-1}$ and l_{ji}, which belong to some alphabet, and traverses position i in w_j. By convention, l_{j0} is the first letter of w_j.

Finding a Word's Segmentation Since a word has an exponential number of possible segmentations[2], it is prohibitive to evaluate all of them in order to find the most likely one. Therefore, the observables in our model are letter

[1] Both morphological analyzers are reference algorithms for the Morpho Challenge.
[2] A word can be segmented in 2^m different ways with $m = n-1$ and n as letter length.

transitions t_{ji} with $Pr(t_{ji}|b_{ji})$ and the hidden variables are the boundary values b_{ji} with $Pr(b_{ji})$ assuming that a boundary in i is inserted independently of other positions. Knowing the parameters of the model, the *letter transition probability distribution* and the *probability distribution over non-/boundaries*, we can find the best segmentation of a given word with $2m$ evaluations using (1).

$$\arg\max_{b_{ji}} Pr(b_{ji}|t_{ji}) = \begin{cases} 1, & \text{if } Pr(b_{ji}=1)\cdot Pr(t_{ji}|b_{ji}=1) \\ & > Pr(b_{ji}=0)\cdot Pr(t_{ji}|b_{ji}=0) \\ 0, & \text{otherwise } . \end{cases} \tag{1}$$

Below, we will define the two parameter distributions explicitly.

Letter Transition Probability Distribution In the Markovian spirit we describe a word by transitions from letters x to y within a morpheme where y is drawn from alphabet A and x from $A_{\mathcal{B}} = A \cup \{\mathcal{B}\}$ where $\mathcal{B}$ is a silent start symbol pointing to the first letter of a morpheme. By introducing such a symbol it is guaranteed that all segmentations of a word have the same number of transitions.

$$\begin{aligned} & p_{x,y} = Pr(X_i = y | X_{i-1} = x) \\ & \text{with } \sum_{y \in A} p_{x,y} = 1 \ \forall x \in A_{\mathcal{B}} \text{ and } 1 \leq i \leq m \ . \end{aligned} \tag{2}$$

Equation (2) is used in (7) and (8) for describing the probability of a letter transition in position i in the PGM.

Probability Distribution over Non-/Boundaries For describing a segmentation we chose a *position-dependent and non-identical distribution*. Each position i is therefore assigned to a Bernoulli random variable Z_i and the existence of a boundary corresponds to a single trial with positive outcome.

$$p_{z_i,m} = Pr(Z_i = 1|m) \tag{3}$$

with $Pr(Z_i = 0|m) + Pr(Z_i = 1|m) = 1$, $1 \leq i \leq m$ and $Z_i \in Z$. The model can be summarised as $\theta = \{X, Z\}$. Equation (3) is subsequently applied to define the probability of segmenting in position i.

Probability of Segmenting in Position i Derived from (3) the probability of putting a boundary in position i is defined as

$$Pr(b_{ji}|m, \theta) = p_{z_i,m} \tag{4}$$

where $p_{z_i,m}$ is the probability of having a boundary value $b_{ji} = z_i$ in i given length m of the segmentation. We rewrite this equation as

$$Pr(b_{ji}|m, \theta) = \prod_{r=0}^{1} (p_{r,m})^{\mu_{b_{ji},r,i,m}} \ , \tag{5}$$

$$\mu_{b_{ji},r,i,m} = \begin{cases} 1, & \text{if } b_{ji} = r \text{ in position } i \text{ given } m \ , \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where we iterate over possible boundary values $r = \{0, 1\}$ and use $\mu_{b_{ji},r,i,m}$ to eliminate all r's in the product which do not correspond to b_{ji} in i given m.

Probability of a Letter Transition in Position i If the segmentation in i is known we can assign a letter transition probability based on (2) and get

$$Pr(t_{ji}|b_{ji}, \theta) = p_{x,y} \tag{7}$$

where transition t_{ji} consists of letter $l_{j,i-1} = x$ and $l_{ji} = y$ given boundary value b_{ji} in position i. For later derivations, we rewrite (7) such that we iterate over the alphabet using x' and y', and eliminate all probabilities which do not correspond to the original x and y using function $\mu_{xy,x'y'}$.

$$Pr(t_{ji}|b_{ji}, \theta) = \prod_{x',y' \in A} (p_{x,y})^{\mu_{xy,x'y'}} \quad . \tag{8}$$

$$\mu_{xy,x'y'} = \begin{cases} 1, & \text{if } x' = x \text{ and } y' = y \ , \\ 0, & \text{otherwise} \ . \end{cases} \tag{9}$$

Finding the Best Segmentation of a Word With (5) and (8) the solution of the problem in (1) becomes

$$b^*_{ji} = \begin{cases} 1, & \text{if } Pr(Z_i = 1|m_j) \cdot Pr(X_i = l_i|X_{i-1} = \mathcal{B}) \\ & > Pr(Z_i = 0|m_j) \cdot Pr(X_i = l_i|X_{i-1} = l_{i-1}) \ , \\ 0, & \text{otherwise} \ , \end{cases} \tag{10}$$

$$b^*_j = (b^*_{j,1}, \ldots, b^*_{j,m}) \ . \tag{11}$$

2.2 Parameter Estimation

Before applying the probabilistic model, its parameters have to be estimated using maximum likelihood estimates or expectation maximization.

Maximum Likelihood Estimates (MLE) We segmented each training set using a heuristic similar to the *successor variety* [8] in a *separate* pre-processing step. All possible substrings of each word were collected in a forward trie along with statistical information, e.g. frequencies. A particular word was decomposed based on probabilities of a letter following a certain substring. From the segmentations we estimated the parameters of Promodes 1 (P1) using MLE.

Expectation Maximization (EM) Parameter estimation by EM [10] was used in Promodes 2 (P2). The EM algorithm iteratively alternates between two distinctive steps, the expectation or E-step and the maximization or M-step, until a convergence criterion is met. In the E-step the log-likelihood of the current estimates for the model parameters are computed. In the M-step the parameters

are updated such that the log-likelihood is maximized. The Q function as the expected value of the log-likelihood function is defined as:

$$Q(\theta, \theta_t) = \sum_{j=1}^{|W|} \sum_{i=1}^{m_j} \sum_{r=0}^{1} \left(Pr(b_{ji} = r | t_{ji}, \theta_t) \log Pr(t_{ji}, b_{ji} = r | \theta) \right) \ , \tag{12}$$

$$\theta^* = \arg\max_{\theta} Q(\theta, \theta_t) \ . \tag{13}$$

The objective function which we want to maximize during the *M-step* is built from the Q function and includes constraints and Lagrange multipliers.[3] The parameters of the model are re-estimated by using partial derivatives which result in the new estimates for the letter transition probabilities as

$$\hat{p}_{x,y} = \frac{\sum_{j=1}^{|W|} \sum_{i=1}^{m_j} \sum_{r=0}^{1} \left(Pr(b_{ji} = r | t_{ji}, \theta_t) \sum_{x',y' \in A} \mu_{xy,x'y'} \right)}{\sum_{y' \in A} \sum_{j=1}^{|W|} \sum_{i=1}^{m_j} \sum_{r=0}^{1} \left(Pr(b_{ji} = r | t_{ji}, \theta_t) \sum_{x'',y'' \in A} \mu_{xy',x''y''} \right)} \ , \tag{14}$$

and for the probability distribution over boundary positions as

$$\hat{p}_{z_i,m} = \frac{\sum_{j=1}^{|W|} \sum_{i=1}^{m_j} \sum_{r=0}^{1} \left(Pr(b_{ji} = r | t_{ji}, \theta_t) \sum_{r'=0}^{1} \mu_{z_i,r',i,m} \right)}{\sum_{r'=0}^{1} \sum_{j=1}^{|W|} \sum_{i=1}^{m_j} \sum_{r=0}^{1} \left(Pr(b_{ji} = r | t_{ji}, \theta_t) \sum_{r''=0}^{1} \mu_{r',r'',i,m} \right)} \ . \tag{15}$$

Although both estimates look complicated, they have an intuitive interpretation. In (14) we count the occurrences of letter transitions from x to y weighted by the posterior probability $Pr(b_{ir}|t_{ji}, \theta_t)$ and divide it by the weighted sum of all transitions from x. In (15) the weighted sum for putting a boundary in i of words with length m is divided by the weighted sum of all boundaries and non-boundaries in i for the same words.

2.3 Committee of Unsupervised Learners

Since different initializations of the EM may converge in different local optima, corresponding models might give slightly different analyses for a single word. PROMODES COMMITTEE (PC) averages results varying initializations and combines them into a single solution using a *committee of unsupervised learners* similar to [12]. A committee can combine results from different algorithms or in our case different initializations. Each committee member can vote for a certain partial or complete solution. The weight of each vote can be uniform or non-uniform, e.g. based on performance or confidence of the algorithm. Our approach is completely unsupervised and purely based on *majority vote* where each vote for putting a boundary in a certain position counts equally. Given η analyses for

[3] The objective function is specified in detail in [11].

a single word w_j in position i we introduce $score_{j,i}$ as

$$score_{j,i} = \sum_{h=1}^{\eta} \pi_{h,j,i} \tag{16}$$

$$\pi_{h,j,i} = \begin{cases} +1, \text{ if analysis } h \text{ contains boundary in } i \text{ of word } w_j, \\ -1, \text{ otherwise} \end{cases} \tag{17}$$

and put a boundary at the ith position of word w_j if $score_{j,i} > 0$.

3 Experimental Results

Although PROMODES is intended for agglutinating languages like Finnish and Turkish, it was also applied to fusional languages like Arabic, German and English. PROMODES decomposes words into their morphemes. Morphosyntactic rules are implicitly stored as statistics in the form of probabilities for segmenting at certain word positions and probabilities for the resulting letter transitions within morphemes. There is no further grammatical analysis like building *signatures* or *paradigms*. Morpheme labels are the morphemes themselves or simple labels consisting of *morpheme[index number]*. The results across languages are listed in Tab. 1 with the highest precision, recall and f-measure written in bold.

General Setup of Experiments Independently of the PROMODES version, we generated a training subset for each language consisting of 100,000 words randomly sampled[4] from each corpus. In the case of Arabic, we employed the entire corpus since it contained less words. For P1 we estimated parameters from the pre-segmented training set which was generated with a simple segmentation algorithm described in Sec. 2.2. By using MLE we averaged statistics across the subset. Subsequently, the model was applied to the entire dataset to decompose all words. P2 used EM to estimate its parameters. Initially, words from the training set were randomly segmented and then the EM algorithm improved the parameter estimates until the convergence criterion was met.[5] The resulting probabilistic model was then applied to the entire dataset. PC made use of the different initializations and resulting analyses of P2. Instead of choosing a single result a committee, described in Sec. 2.3, combined different solutions into one.

Analysis of Results In general, PROMODES performed best on non-vowelized (nv.) and vowelized (vw.) Arabic, well on Finnish and Turkish, and moderately on English and German compared to other approaches in the Morpho Challenge 2009. For a detailed comparison see [14]. Of the three PROMODES versions there was no best method for all languages. An analysis of the different gold

[4] No frequency or word length considerations.

[5] We used the *Kullback-Leibler divergence* [13] which measures the difference between the model's probability distributions before and after each iteration of the EM.

standards suggested, however, that all PROMODES methods performed better on languages with a high morpheme per word ratio. In detail, the best result (f-measure) for English was achieved by P1, for Arabic (nv./vw.), German and Turkish by P2, and for Finnish by PC. Especially for nv. Arabic, PROMODES achieved a high precision (at the cost of a lower recall). This implies that most morphemes returned were correct but only few were found. The reason for that might be that words were quite short (5.77 letters on av.)[6] and lacking the grammatical information carried by the vowels. Furthermore, words contained more morpheme labels per word (8.80 morphemes on av.) than letters which made morpheme analysis challenging. PROMODES showed better results on vw. Arabic which were also more balanced between precision and recall. Especially for English with longer words (8.70 letters on av.) and fewer morphemes per word (2.25 morphemes on av.), PROMODES exhibited a different behavior with a low precision but a high recall. This suggests that the algorithm splits words into too many morphemes. A similar effect was encountered for the morphologically more complex languages Finnish and Turkish where PROMODES tended to oversegment as well. For German, precision and recall varied a lot with different PROMODES versions so that a general pattern could not be identified.

Table 1. Results of P1, P2, PC in Competition 1

Language	Precision			Recall			F-measure		
	P1	P2	PC	P1	P2	PC	P1	P2	PC
Arabic (nv)	**.8110**	.7696	.7706	.2057	**.3702**	.3696	.3282	**.5000**	.4996
Arabic (vw)	**.7485**	.6300	.6832	.3500	**.5907**	.4797	.4770	**.6097**	.5636
English	**.3620**	.3224	.3224	**.6481**	.6110	.6110	**.4646**	.4221	.4221
Finnish	.3586	.3351	**.4120**	.5141	**.6132**	.4822	.4225	.4334	**.4444**
German	**.4988**	.3611	.4848	.3395	**.5052**	.3461	.4040	**.4212**	.4039
Turkish	.3222	.3536	**.5530**	**.6642**	.5870	.2835	.4339	**.4414**	.3748

4 Conclusions

We have presented three versions of the PROMODES algorithm which is based on a probabilistic model. The parameters of PROMODES 1 (P1) were estimated using maximum likelihood estimates. Expectation maximization was applied in PROMODES 2 (P2). PROMODES COMMITTEE (PC) combined results from different initialisations of P2 by using a committee of unsupervised learners. All three methods achieved competitive results in the Morpho Challenge 2009. The strengths of PROMODES, in general, are that it does not make assumptions about the structure of the language in terms of prefix and suffix usage. Furthermore, instead of building a morphological dictionary and a rule base which are likely to be incomplete, it applies statistics of a small training set to a larger test

[6] Average measures based on the respective gold standard.

set. This is achieved at the cost of over-segmenting since there is no inductive bias towards a compressed morphological dictionary. Our future work includes extending the probabilistic model to a higher order which should increase the model's memory and therefore reduce over-segmentation. We also intend to further analyse the behaviour of the committee and examine the impact of different training set sizes.

Acknowledgement

We would like to thank Aram Harrow for fruitful discussions on the mathematical background of this paper, our team colleagues Roger Tucker and Ksenia Shalonova for advising us on general issues in morphological analysis and the anonymous reviewers for their comments. This work was sponsored by EPSRC grant EP/E010857/1 *Learning the morphology of complex synthetic languages.*

References

1. Booij, G.: The Grammar of Words: An Introduction to Linguistic Morphology. Oxford University Press (2004)
2. Goldsmith, J.: Segmentation and Morphology. In: The Handbook of Computational Linguistics. Blackwell (2009)
3. Goldsmith, J.: Unsupervised learning of the morphology of a natural language. Computational Linguistics **27** (2001)
4. Monson, C.: ParaMor: From Paradigm Structure To Natural Language Morphology Induction. PhD thesis, Carnegie Mellon University, Pittsburgh, PA (2008)
5. Creutz, M., Lagus, K.: Inducing the morphological lexicon of a natural language from unannotated text. Proc. of AKRR **1** (2005)
6. Bernhard, D.: Simple morpheme labeling in unsupervised morpheme analysis. Working Notes for the CLEF Workshop, Hungary **1** (2007)
7. Dang, M.T., Choudri, S.: Simple unsupervised morphology analysis algorithm (sumaa). Proc. of PASCAL Workshop on Unsuperv. Segmentation of Words into Morphemes, Italy **1** (2006)
8. Harris, Z.S.: From phoneme to morpheme. Language **31** (1955)
9. Snover, M.G., Brent, M.R.: A probabilistic model for learning concatenative morphology. Proc. of NIPS (2002)
10. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithms. Journal of the Royal Statistical Society **39** (1979)
11. Spiegler, S., Golenia, B., Flach, P.: Promodes: A probabilistic generative model for word decomposition. Working Notes for the CLEF 2009 Workshop, Greece (2009)
12. Atwell, E., Roberts, A.: Combinatory hybrid elementary analysis of text (cheat). Proc. of PASCAL Workshop on Unsuperv. Segmentation of Words into Morphemes, Italy **I** (2006)
13. Kullback, S., Leibler, R.A.: On information and sufficiency. Annals of Mathematical Statistics **22** (1951)
14. Kurimo, M., Virpioja, S., Turunen, V.T., Blackwood, G.W., Byrne, W.: Overview of morpho challenge 2009. In: Multilingual Information Access Evaluation Vol. I, CLEF 2009, Greece, Lecture Notes in Computer Science, Springer. (2010)