HIDDEN MARKOV MODELS FOR INDUCTION OF MORPHOLOGICAL STRUCTURE OF NATURAL LANGUAGE

Hannes Wettig, Suvi Hiltunen and Roman Yangarber

Department of Computer Science,
University of Helsinki, Finland
First.Last@cs.helsinki.fi

ABSTRACT

This paper presents initial results from an on-going project on automatic induction of morphological structure of natural language, from plain, un-annotated textual corpora. In previous work, this area has been shown to have interesting potential applications. One of our main goals is to reduce reliance on heuristics as far as possible, and rather to investigate to what extent the morphological structure is *inherent* in the language or text per se. We present a Hidden Markov Model trained with respect to a two-part code cost function. We discuss performance on corpora in highly-inflecting languages, problems relating to evaluation, and compare to results obtained with the *Morfessor* algorithm.

1. INTRODUCTION

In this paper we present our work on automatic induction of morphological structure of natural language. Our interest is in languages that exhibit interesting and complex morphological phenomena. Our ultimate goal is to understand which of these phenomena may be discovered automatically in an unsupervised fashion. The question is whether the complete morphological system can be discovered automatically from plain, natural-language text, or to what extent it can be discovered automatically. This implies the weaker question, whether the morphological system is somehow "inherently encoded" in the language or in the corpus itself.

Several approaches to morphology learning and induction have emerged over the last decade, summarized in section 2.1. In the prior work, it has been observed that morphology induction has interesting potential applications, among them the possibility for rapidly building morphological analyzers for resource-poor languages, including slang.

Our goals and concerns are somewhat more theoretical, at least initially; we are at present less interested in applications than in building models that are principled and avoid building ad-hoc heuristics into the models from the outset.

In the following sections, we present a statement of the morphology induction problem, Section 2, review prior work in Section 2.1, present our model in Section 3, and evaluation in Section 4.

2. MORPHOLOGY INDUCTION

Our work focuses on highly inflecting languages. We have so far experimented with corpora in Finnish and Russian. Finnish is highly inflecting, its morphology traditionally called "agglutinative," although it exhibits a wide variety of flexion, morpho-phonological alternation and vowel harmony. Finnish has a rich system of derivation, inflection, and productive and complex nominal compounding, where multiple elements of the compound may be inflected.

For example: *talvikunnossapito* "winter-time in-shape-keep" lit. *talvi # kunto + ssa # pito* = "winter # shape + in # keeping", where '#' is the compound boundary, '+' is a morpheme boundary, and 'in' is a marker of inessive case (the locative case in Finnish, indicating presence in a location or being in a state). In Finnish, derivation and inflection are achieved via suffixation, and prefixation is practically unavailable. Russian exhibits complex morphology, similar to most Slavic languages (except for some South Slavic languages that have dropped nominal morphology); compounding is limited and mostly not productive, but there is rich prefixation.

We also note that these languages use relatively recent writing systems, which allows us to ignore potential discrepancies between written and spoken representations, and we make the simplifying assumption that they are the same.

Our ultimate goal is to model all aspects of morphology, including classification of morphemes into meaningful morphological categories, and capturing allomorphy or morpho-phonological alternation; one seminal current approach to morphological description is the two-level paradigm, [1]. Initially, as is done in other prior work, we try to build a solid baseline model and focus first on obtaining a good *segmentation*; once we have a way of segmenting words into morphs, or morph candidates, we plan to model the more complex morphological phenomena.

2.1. Prior Work

There was active research in unsupervised morphology induction, especially during the decade from 2000-2009, we mention only a few here.

Our work is closely related to a series of papers by Creutz and Lagus, starting with [2, 3]. Unlike some of their work, e.g., [4], we do not posit *a priori* morphological classes; we aim to allow the model to learn the classes and distribution of morphs automatically without heuristics. Like Creutz and Lagus's work, ours uses MDL (the Minimum Description Length principle, see, e.g., [5]) as a measure of goodness.

Introduced earlier, a rather different MDL-based morphology learner was Linguistica, [6]. In Linguistica, an essential feature is the notion of *signatures*, to describe morphs belonging to morphological classes consisting of affixes describing a morphological paradigm, e.g., nominal suffixes for a given declension class, verbal suffixes, etc. Another approach somewhat similar to ours is pursued in [7]. Their model is stated as a Finite State Automaton—effectively equivalent to an HMM—but is less general and once more employs different learning heuristics, which would make this approach fail for languages of richer morphological structure.

There is a range of more distant theories and approaches to morphology induction, stimulated in part by the natural observation that children are able to learn morphological structure at a very early age as part of language acquisition—i.e., using very little data—which makes such acquisition by machine a fascinating challenge in itself.

3. OUR METHOD

Morphology of many languages is commonly modeled as a finite state network, with the states corresponding to morphological classes. A state/class may correspond to a class of nouns or verbs belonging to a certain paradigm, or a set of suffixes, or prefixes belonging to a certain paradigm. The main idea of the model is to discover, for every word in a corpus, the sequence of states that will generate the word.

In the rest of the paper, we will write *morphology* to mean a *segmentation* of a corpus or a word list into "morphs"—meaningful segments—and a *classification*, the assignment of the morphs to meaningful morphological classes. As noted before, this is not strictly correct, since there is a great deal more to morphology than segmentation and classification, but we will use this terminology for the present.

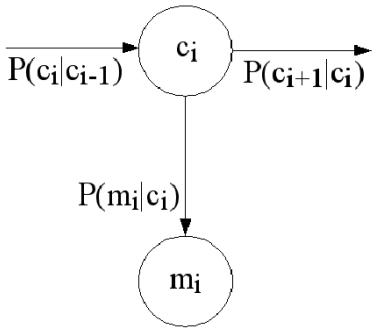

Figure 1. The HMM with hidden *class* states and observed *morph* states.

3.1. The Hidden Markov Model

The model is depicted in Figure 1 and described by:

- *Lexicon*: a set of morphological classes. Each class is a set of morphs. A morph may belong to more than one class. Each class corresponds to a *state* in the HMM.
- *Transition probabilities*: the probability of transitioning from one class to another.
- *Emission probabilities*: the probability of generating a morph from a class, given that the model is in the state corresponding to the class.

The model consists of a set of states C_i, which generate morphs with certain *emission* probabilities, and transition probabilities between pairs of states. For convenience, we also include a special starting state C_0 and final state C_F. The starting state generates nothing (or, always an empty string), and the final state emits the final word boundary ('#') with probability 1. The states should, in principle, correspond to true morphological classes, e.g., the class of all noun stems falling under a certain paradigm, or

the class of all suffixes for a given nominal paradigm. The former is an example of an open (i.e., potentially very large) class, whereas the latter is a closed (very small) class. We model all classes in the same way; a different approach is taken in, e.g., [3, 6].

3.2. MDL Cost

The MDL cost of the *complete* data under the model is the sum of the costs of coding the lexica, the transition, and the emission probabilities.

$$Cost = Lex + Tran + Emit$$

Lexicon: To code the lexica, for each class C_i, we simply encode the strings in the lexicon one by one:

$$Lex = \sum_i \left[\sum_{m \in C_i} L(m) - \log |C_i|! \right] \qquad (1)$$

i ranges from 1 to K, the number of classes, and m ranges over all morphs in class C_i; the number of morphs in class C_i is denoted $|C_i|$. $L(m)$ is a prefix code-length for morph m, in the current implementation simply $L(m) = \log(|\Sigma| + 1) \cdot (|m| + 1)$, where Σ is the alphabet and $|m|$ is the number of symbols forming morph m. This code is somewhat wasteful, and we plan to use a tighter code in the future. The term $-\log |C_i|!$ accounts for the fact that we do not need to code the morphs of the lexicon in any specific order.

Transitions and Emissions: We code the data given the lexica—namely the paths of class transitions from C_0 to C_F, from word start to finish—prequentially [8], using Bayesian Marginal Likelihood. We employ uniform priors.

Ideally, we would have preferred to use the *normalized maximum likelihood* (NML) [9], but it is unclear how to calculate it for two reasons. First, the model an HMM, for which no efficient method to calculate the *regret* is known; second, the data size—the number of *tokens*, i.e., instantiations of morphs—varies during the search for a good segmentation. Therefore it is unclear what the regret would be in this setting.

3.3. Search

We start with a random segmentation and random classification (assignment of tokens to classes). We choose the number of classes to be $K = 100$, larger than what we would expect in a "true" morphology, to assure that the model is sufficiently expressive. We then greedily re-segment each word in the corpus, minimizing the total code-length.

Our morphology induction algorithm:

1. *Input*: A large list of words in the language.
2. *Initialize*: Create a random initial segmentation and class assignment.
3. *Re-segment*: For each word find the best segmentation with respect to the two-part code-length, given the current data counts, as described in Section 3.4.
4. *Repeat*: Step 3 until convergence.

3.4. Re-Segmentation

We now explain how we compute the most probable segmentation of a word into morphs, given a set of transition and emission probabilities. The logarithm of any transition or emission probability corresponds to change in code-length that is induced by the increment in the corresponding count. We apply a Viterbi-like search algorithm for every word w in the corpus, to compute the most likely path through the HMM, given the word, *without* knowledge of the morphs that cover the word. Standard Viterbi would only give us the best class assignment *given* a segmentation. The search algorithm fills in the matrix below, using dynamic programming, starting from the leftmost column toward the right:

Class	σ^1	σ^2	...	σ^j	...	$\sigma^n = w$	#
C_1							
C_2							
...							
C_i				X			
...							
C_K							

The notation we will use: σ_a^b is a *substring* of w, from position a to position b, inclusive; positions are numbered starting from 1. We will use the shorthand $\sigma^b \equiv \sigma_1^b$ (a prefix of w up to b) and $\sigma_a \equiv \sigma_a^n$ (a suffix). A single *morph* μ_a^b lies between positions a and b in w. Thus σ_a^b is just a sub-string, and may contain several morphs, or cut across morph boundaries.

In the cell marked X, we compute $P(C_i|\sigma^j)$, the probability that the HMM is in state C_i given that it has generated the prefix of w up to the j-th symbol. This probability is computed as the maximum

over the following expressions, using values already available from columns to the left:

$$P(c_i|\sigma^j) = \max_{q,a} P(C_q|\sigma^a) \cdot P(C_i|C_q) \cdot P(\mu_{a+1}^j|C_i) \quad (2)$$

where the maximum is taken over $q = 0, 1, \ldots, K$, and $a = j-1, j-2, \ldots, 0$, i.e., q ranges over all states, and a ranges over the preceding columns; here $P(\mu_{a+1}^j|C_i)$ is the probability of emitting the string μ_{a+1}^j as a single *morph* in state i, for some $a < j$. For the empty string, $\sigma^0 \equiv \epsilon$, we set $P(C_q|\sigma^0) \equiv 1$ if $C_q = C_0$, the initial state, and zero for all other states.

The transition to the final state C_F is computed in the rightmost column of the matrix, marked #, using the transition from the last morph-emitting state—from column σ^n—to C_F. (State C_F emits the word boundary # with probability 1). Thus, the probability of the most likely path to generate w is:

$$\max_q P(C_q|\sigma^n) \cdot P(C_F|C_q) \cdot P(\#|C_F)$$

where the last factor $P(\#|C_F)$ is always 1.

In addition to storing $P(C_i|\sigma^j)$ in the cell (i, j) of the matrix, we store also the "best" (most probable) state q from which we reached this cell, and the column a from which we arrived at this cell. These values, the previous state (row) and column, allow us to backtrack through the matrix at the end to reconstruct the most probable—lowest-cost—path through the HMM.

4. EVALUATION

Evaluation of morphology discovery is a complicated matter, particularly when morphological analysis is limited to *segmentation*—as it is in our current work and most prior work—mainly because in general it is not possible to posit definitively "correct" segmentation boundaries which by definition ignore information about allomorphy.

One evaluation scheme is suggested in the papers about the HUTMEGS "Gold-standard" evaluation corpus for Finnish, [10, 11]. We have two concerns with the evaluation suggested in HUTMEGS.

Consistency: [10] observe that insisting on a single "correct" morphological analysis for a word form is not possible. A motivating example, in English, *tries* can be analyzed as *tri+es* or *trie+s*; the "correct" analysis is that there are two morphs, a stem and a suffix, and each has more than one allomorph. Restricting morphological analysis to segmentation makes the problem ill-defined: we cannot posit a "proper" way to place the morpheme boundary, and also expect an automatic system to discover this particular way.

The HUTMEGS approach, [10], proposes "fuzzy" morpheme boundaries, to allow the system free choice within the bounds of the fuzzy boundary—as long as the system splits the word somewhere inside the boundary, it is not penalized for an incorrect segmentation.

A problem with this approach is that it is too permissive: the system should commit to a certain way of segmenting similar words, and then consistently segment according to its "theory"—it should then be penalized for violating its own decision by placing the boundaries *differently* in similar cases.

Sensitivity: there is a difference between inflectional and derivational morphological processes, and we believe it should be taken into account during evaluation. Specifically, inflectional morphology is much more transparent and productive. By contrast, derivation is much more opaque. The boundaries can be very unclear between fully productive derivation and processes that may have been productive in the past, but have ceased to be productive, or may have resulted in fixed or ossified forms.

For example, the verbal stem *menehty-*, "perish", is derived from the stem *mene-*, "go" (similarly to "pass away"); the morphs *-ht-y-* are productive derivational suffixes in Finnish. However, the native speaker feels no direct intuitive link between the two stems: the derived form ossified into a separate meaning too long ago, and is not perceived as an instance of productive derivation.

In light of such variation of morphological processes, it seems to make sense a. to mark the inflectional and derivational boundaries *differently* in the gold standard, and b. to penalize the system differently for missing inflectional boundaries vs. derivational boundaries.

4.1. Data

In our initial experiments we use data from Finnish and Russian. The Finnish corpus consists of the 50,000 distinct words in the 1938 translation of the Bible, from the Finnish Corpus Bank. The Russian corpus contains 70,000 distinct words from of five novels by Tolstoy, downloaded from an on-line library. The corpora were pre-processed to remove all punctuation and obtain lists of distinct words.

For each corpus, we selected several random sam-

ples, 100 words each, for gold-standard annotation and evaluation. We experimented with two kinds of samples: a *mixed* sample is a purely random collection of 100 words; a *chunk* sample is a list of lexicographically consecutive words, with a random starting point. We used chunks of words because we wanted to see whether the model performs consistently on very similar and related words.

Three annotators, marked the gold-standard annotation for the samples, based on their linguistic intuition, without reference to segmentations generated by the system. In the gold standard, the derivational and inflectional boundaries were marked differently, but to simplify the initial evaluation, we make no distinction between derivational and inflectional boundaries at present.

	Precision	Recall	F-measure
Finnish		*Chunk*	
Greedy .20	**0.368**	**0.352**	**0.360**
Greedy .25	0.357	0.323	0.339
Morfessor	0.286	0.237	0.259
		Mixed	
Greedy .20	0.540	0.364	0.435
Greedy .25	0.472	0.337	0.393
Morfessor	**0.627**	**0.408**	**0.494**
Russian		*Chunk*	
Greedy .20	**0.533**	**0.484**	**0.508**
Greedy .25	0.468	0.4	0.431
Morfessor	0.463	0.392	0.425
		Mixed	
Greedy .20	**0.622**	**0.38**	**0.472**
Greedy .25	0.508	0.31	0.385
Morfessor	0.570	0.346	0.431

Table 1. Evaluation against gold standard on four samples of words.

4.2. Experiments

Results of the experiments with our HMM algorithm are given in Table 1 in terms of recall, precision, and F-measure (which combines the two). The Morfessor algorithm, described in [10], was run on the same data for comparison. The conditions we experimented with are as follows.

Parameter $\rho = 0.20$ and 0.25 is the probability of placing a morph boundary between any two adjacent symbols during the initial segmentation.

The algorithm was run to convergence. An example of the convergence curve of the MDL cost for the Finnish text is in figure 2. We can make the following observations.

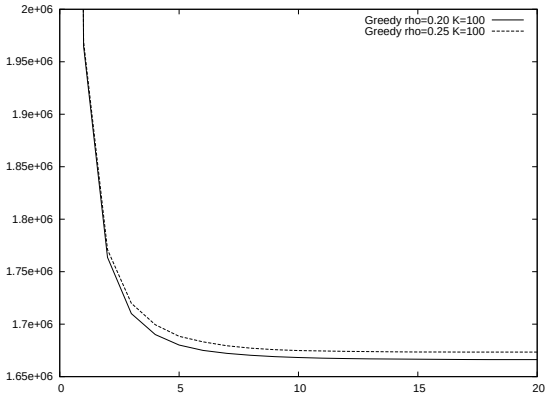

Figure 2. MDL cost convergence.

The results obtained with our method compare favourably with those obtained by Morfessor—the best competitor for this task that we are aware of—with the exception of the sample *Finnish/Mixed*. Precision is generally better than recall, which suggests our algorithm places too few morph boundaries. Note, our gold standard does not use the fuzzy approach, allowing fuzzy boundaries, which would make the system free to place a boundary anywhere within a span of several symbols; such an approach would yield artificially optimistic performance numbers.

It is important to note the large variance in performance (in terms of both recall and precision) of our algorithm with different but fairly close values of the initial ρ parameter, 0.20 vs. 0.25. This clearly indicates that the current implementation gets stuck in local optima fairly quickly, as seen from Figure 2.

Another indication of the same problem comes from visual observation of the *Chunk* data samples: we sometimes observe that within a set of consecutive, very similar words (members of the same paradigm), words are segmented *differently*, some correctly and others not.

5. CONCLUSIONS AND CURRENT WORK

We have introduced a Hidden Markov Model to automatically segment a corpus of natural language, while grouping the discovered morphs into classes of appearance at similar places in the words of the corpus. This problem is actively researched, but we believe our proposed model approaches it in a more general and systematic way. Using no prior knowledge about the language in question, we start from a randomly

initialized model and train on the corpus by optimizing a two-part code-length function. We believe that the preliminary results obtained with a rudimentary coding scheme coupled with a simple greedy search are promising.

Several improvements are currently under construction. For the cost function, a natural improvement is coding the lexica more efficiently. Taking into account the letter frequencies should lead to better results. Another issue to be addressed is automatic adjustment of the number of classes, which should be reflected in the code-length.

Equally simple enhancements can and will be done for the search. Greedy search, as it stands, quickly converges to local—far from global—optima. Strategies to improve this situation include:

- switching from Greedy search to Simulated Annealing.
- expanding the neighbourhood to moving *morphs* from one class to another (as opposed to just *tokens*).
- switching from Greedy search to EM (Expectation-Maximization). This demands calculating the *expected* segmentation, i.e. a weighted sum over *all* segmentations. We believe this to be possible, though computationally demanding. Fortunately, word-by-word analysis is easily parallelized.

An adequate evaluation scheme remains a serious problem, we point out two shortcomings of a state-of-the-art approach to evaluation—consistency and sensitivity. We suggest an improvement by building additional information into the gold standard in a principled fashion.

Segmentation of words into morphs and morph classification gives us a good baseline analysis. We next intend to work on open problems, focusing especially on those aspects of linguistic structure that our relatively simple HMM cannot describe. The presence of allomorphy currently results in an overly complex HMM. This happens because simple rules determining the choice of an appropriate allomorph cannot be expressed by the model—instead, the allomorphic variants of a morph will be categorized into different classes, depending on how they interact with nearby morphs. This calls for a *context model*, of which we expect to considerably reduce the code-length.

6. REFERENCES

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

[2] M. Creutz and K. Lagus, "Unsupervised discovery of morphemes.," in *Proc. Wkshop. on Morphological and Phonological Learning*, Philadelphia, PA, USA, 2002.

[3] M. Creutz, "Unsupervised segmentation of words using prior distributions of morph length and frequency.," in *Proc. 41st Meeting of ACL*, Sapporo, Japan, 2003.

[4] M. Creutz, "Induction of a simple morphology for highly-inflecting languages.," in *Proc. ACL SIGPHON)*, Barcelona, Spain, 2004.

[5] P. Grünwald, *The Minimum Description Length Principle*, MIT Press, 2007.

[6] J. Goldsmith, "Unsupervised learning of the morphology of a natural language," *ACL*, vol. 27, no. 2, 2001.

[7] J. Goldsmith and Y. Hu, "From signatures to finite state automata," in *Midwest Computational Linguistics Colloquium*. Bloomington, IN, 2004.

[8] A.P. Dawid, "Statistical theory: The prequential approach," *J Royal Statistical Society, A*, vol. 147, no. 2, 1984.

[9] Y. Shtarkov, "Universal sequential coding of single messages," *Problems of Information Transmission*, vol. 23, 1987.

[10] M. Creutz and K. Lindén, "Morpheme segmentation gold standards for finnish and english," Technical Report A77, HUT, 2004.

[11] M. Creutz, K. Lagus, K. Lindén, and S. Virpioja, "Morfessor and hutmegs: Unsupervised morpheme segmentation for highly-inflecting and compounding languages.," in *Proc. 2nd Baltic Conf. on Human Language Technologies*, Tallinn, Estonia, 2005.