A Multi-purpose Bayesian Model for Word-Based Morphology

Maciej Janicki

University of Leipzig, Institute of Computer Science
Augustusplatz 10, 04109 Leipzig, Germany
`janicki@informatik.uni-leipzig.de`

Abstract. This paper introduces a probabilistic model of morphology based on a word-based morphological theory. Morphology is understood here as a system of rules that describe systematic correspondences between full word forms, without decomposing words into any smaller units. The model is formulated in the Bayesian learning framework and can be trained in both supervised and unsupervised setting. Evaluation is performed on tasks of generating unseen words, lemmatization and inflected form production.

Keywords: word-based morphology, machine learning, generative model, inflection, lemmatization, lexicon expansion

1 Introduction

Morphological analysis is an indispensable element of the NLP pipeline for many languages. Lemmatization or stemming is essential for virtually every semantic processing and information retrieval task, whereas syntactic processing, like parsing or chunking, usually requires Part-of-Speech tags and inflectional features, e.g. case or gender. As a rich inflectional system is typically able to generate hundreds of word forms for a single lemma, storing all those information in a lexicon is highly inefficient. In addition to inflection, also derivational morphology (especially compounding is some languages, e.g. German) often employs highly productive and regularized processes, which can result in a potentially unlimited number of lemmas. Therefore, systems for automatic morphological processing are a topic of ongoing research.

Despite the importance of morphological analysis, up to now no clear task definition has been established. The output of tools ranges from morpheme segmentation and labeling to just inflectional analysis (lemma + tag). Although the segmentation-based approach provides more detailed information, some non-straightforward tasks are often left to the user, like distinguishing between inflectional and derivational affixes, reconstructing the lemma or deriving the properties of the word from the properties of its morphemes. Also the handling of non-concatenative morphological phenomena varies from tool to tool. The reasons for those problems lie already in the underlying morphological theory, which requires all word formation processes to be expressed in terms of morphemes.

Therefore, we introduce alternative theories, in particular the one called Whole Word Morphology, in Sect. 3.

In addition to morphological analysis, there are also other NLP tasks, which require the knowledge of a language's morphology. *Inflected form generation*, understood as producing an inflected word form from a lemma and a set of desired inflectional features, is needed in machine translation, among others. Another morphology-related task is *lexicon expansion*, i.e. anticipating morphologically motivated, but unseen words.

The goal of the present work is a generative probabilistic model of the lexicon, that accounts for morphological relations between words. It can be trained either in an unsupervised or a supervised setting and, once trained, it is capable of solving various morphology-related tasks, including the above-mentioned. Relying on a relational description of morphology, it does not suffer from the limitations of the segmentation-based approaches.

The rest of this paper is structured as follows: Section 2 provides an overview of the state-of-the-art in machine learning of morphology. Section 3 introduces linguistic theories, on which the present work is based. The generative model is described in Sect. 4 and the algorithms for training and applying it are sketched in Sect. 5. Section 6 describes evaluation of the model on practical NLP tasks. Topics for further research are described in Sect. 7. Section 8 summarizes the opportunities and advantages of the present work.

2 Related Work

2.1 Morpheme Segmentation

Automatic morpheme segmentation, especially unsupervised, has been a topic of active research for at least the last two decades [10]. The probably most known state-of-the-art tool is Morfessor [23]. It is based on the Minimum Description Length principle, which in this case is strongly connected to Bayesian learning. Other approaches based on probabilistic models include the work of Poon et al. [16] (log-linear models) and Can [5] (chap. 5, probabilistic hierarchical clustering).

Another group of approaches seeks to first group morphologically similar words together. The words belonging to the same cluster are then aligned in order to extract morphemes. Especially context similarity is typically used for this purpose, sometimes along with ortographical similarity. This group includes the approaches of Can [5] (chap. 4) and Kirschenbaum [12], among others.

Many further approaches have been submitted to MorphoChallenge [13], which offered standarized task formulation and datasets for morpheme segmentation. This yearly competition took place from 2005 to 2010.

Supervised learning of morphological segmentation is significantly less common. A method using Conditional Random Fields has been presented recently by Ruokolainen et al. [19]. Moreover, some models developed for unsupervised learning can also be trained in a supervised setting, like Morfessor [23] or the log-linear model of Poon et al. [16].

2.2 Lemmatization and Learning Inflectional Paradigms

Another branch of research considering automatic learning of morphology targets especially inflectional morphology. The task is then defined as either aligning an inflected word form to its root or lemma, or clustering the inflected forms of same lemma together. Many approaches exploit the high mutual dependency of inflectional processes, expressed in *paradigms*, which is a characteristic feature of inflection.

Yarowsky and Wicentowski [26], followed by Wicentowski [25], present a model of inflection, which can account for some non-concatenative phenomena, like root vowel change. The supervised approach employs a trie-based classifier, whereas the unsupervised performs the matching between an inflected word form and lemma based on a combination of various features: ortographical similarity, context similarity and frequency ratio. The paradigm-oriented approaches include Chan [6], who uses Latent Dirichlet Allocation to group suffixes into paradigms, and Durrett and DeNero [8]. Unsupervised clustering of inflected word forms into lexemes has been approached by Janicki [11].

2.3 Tagging Unknown Words

A slightly different approach to morphological analysis is found in handling unknown words in the task of stochastic PoS-tagging. In this case, the goal is to predict the tag and sometimes the lemma of an unknown word in a given context. The tagging is thus token-based, rather than type-based. In addition to context features, which are the main component of such taggers, morphological features like prefix and suffix n-grams are sometimes incorporated in order to improve the tagging. The examples of such approaches include Mikheev [14], Tseng et al. [22] and Chrupała et al. [7], among others.

2.4 Lexicon Expansion

Compared to the above topics, the prediction of morphologically motivated unseen words is relatively little explored. Rasooli et al. [18] have shown in a recent paper, how the segmentation produced by Morfessor can be used for generation of new words. Their approach is to generate all possible sequences of morphemes with a finite-state automaton and to apply additional reranking steps based on letter trigram probabilities. On the other hand, Neuvel and Fulop [15] define the whole task of morphology learning as learning to produce new words using morphological mechanisms. The approach is inspired by the theory of Whole Word Morphology [9] and bears many similarities to the present work. However, it does not use probability or any kind of scoring, the discovered rules are applied wherever possible, which may lead to overgeneration, especially when learning from noisy data.

3 Word-Based Morphology

The notion of *word-based morphology* has been introduced by Aronoff [2] with the claim that "all regular word-formation processes are word-based", i.e. they apply to whole existing words, rather than some abstract structural elements. While Aronoff only took derivational morphology into account, the claim was further extended by Anderson [1]. In the latter theory, both inflection and derivation consists of word-formation rules, which operate on *stems* (defined as "word minus inflectional material") without creating any internal word structure. The difference is that inflectional rules derive surface words out of stems, while derivational rules derive new stems.

A theory that rejects any abstract elements of word structure, called *Whole Word Morphology* (henceforth WWM), has been proposed by Ford et al. [9]. A lot of criticism is devoted to the notion of "morpheme" there: first of all, it does not account for non-concatenative morphological phenomena. Also, the definition of morpheme as "the minimal element of language having a meaning or function" is troublesome, since some units participating in word-formation processes do not have a meaning on their own, while some functions are realized not by addition, but rather by absence or even truncation of phonological material. An example would be French adjective inflection, where the masculine form is formed from the feminine by truncation of the last consonant. Finally, the distinction between inflection and derivation is rejected as unmotivated.

In WWM, the minimal meaningful elements of language are words themselves. Morphology, on the other hand, describes the frequently recurring formal similarity patterns between words in terms of rules, called *morphological strategies*. Those rules always relate full lexical representations (phonological, syntactic and semantic) of two words to each other and are not decomposable. An example of a rule for English plural formation would be $/X/_{\text{N.Sg}} \leftrightarrow /Xs/_{\text{N.Pl}}$. X is here a variable element, that can be instantiated with any string of phonemes. The two-sided arrow indicates, that the rule is bidirectional: no direction is privileged and no word is said to be morphologically "more complex" than the other. The atomicity of the rule means that no direct link is established between the *-s*-ending and the Pl feature. In other words, there is no "plural morpheme", there is just a systematic correspondence between many plural nouns and their singular counterparts. While such interpretation may seem awkward in this case, it allows to treat all morphological phenomena uniformly, including the above mentioned non-concatenative and truncation cases, among others.

Finally, WWM does not distinguish "compounds" as words derived from more than one other word. It is pointed out, that only one part is always responsible for the base meaning and grammatical properties of a compound, while the other part is merely an "affix", the similarity of which to an existing word being irrelevant for the morphology. Compounds can thus also be explained with regular morphological strategies: for example the German word *Arbeitsmarkt* is linked to *Markt* via the rule: $/X/_{\text{N}} \leftrightarrow /arbeitsX/_{\text{N}}$. Doubtful cases between derivation and compounding, like the German rule $/X/_{\text{Adj.Pred}} \leftrightarrow /Xerweise/_{\text{Adv}}$, speak in favour of this unification.

4 The Model

4.1 Lexicon as Directed Graph

The model of morphology introduced in this paper adopts many of the ideas from the WWM theory. The morphological structure of a language is understood as a graph of words, with morphological rules as edges. However, the bidirectionality of the rules is not preserved, because it would lead to many redundant edges. Instead, every word can only be derived from a single base word, i.e. every node can have at most one ingoing edge.

$$\begin{tikzcd}
 & \text{machst} & \\
\text{machen} \arrow[ur] \arrow[r] \arrow[dr] \arrow[ddr] & \text{macht} & \\
 & \text{machte} \arrow[r] & \text{machtest} \\
 & \text{machbar} \arrow[r] \arrow[dr] & \text{machbaren} \\
 & & \text{machbare}
\end{tikzcd}$$

Fig. 1. A sample tree from a German lexicon.

Figure 1 presents a sample graph for German lemmas *machen* 'to do' and *machbar* 'feasible' with a couple of inflected forms. Note that if we used bidirectional rules, an edge would have to be drawn between every pair of the shown words, also pairs like (*machtest*, *machbaren*). Such a multitude of redundant rules can hardly correspond to a language speaker's competence. Therefore, we are rather looking for a spanning tree of the full morphological graph, which contains for each word a single base word, from which it is "really" derived.

In Fig. 1, the edges are drawn according to the usual morphological theory: inflected forms are derived from lemmas (possibly through other inflected forms, like *machtest* from the "imperfect stem" *machte*), while derivational rules derive "more complex" lemmas from "simpler" ones. Such behavior might be desirable for solving some specific tasks, like lemmatization, and can be controlled by training data and model parameters. However, from the point of view of model architecture, *any* trees of words are allowed, regardless of whether they make sense for the linguistic theory. Especially in the unsupervised learning task, where the graph structure emerges from the data through optimization, the regularities in word shape can be captured in quite different ways than traditional grammatical description (see also introduction to sec. 6).

4.2 Model Formulation

A Bayesian model with parameter θ consists of two probability distributions: the *prior probability* $P(\theta)$, which expresses a-priori beliefs about the possible parameter values, and the *likelihood function* $P(D|\theta)$, which expresses the compatibility of the data D with the model parametrized by θ. The goal of the learning task is to find the most likely model given the observed data, i.e. the parameter value θ^* which maximizes the *posterior probability* $P(\theta|D)$. The latter can be transformed using the Bayes' theorem:

$$\theta^* = \arg\max_{\theta} P(\theta|D) = \arg\max_{\theta} \frac{P(\theta) \cdot P(D|\theta)}{P(D)} = \arg\max_{\theta} P(\theta) \cdot P(D|\theta) \quad (1)$$

The last equality follows because $P(D)$ does not depend on θ. Instead of computing the likelihood directly, the *log-likelihood* is often used, because it is easier to manipulate numerically. As logarithm is an increasing function, the maximization of those two is equivalent.

In our model of morphology, the observed data is the *lexicon* L: a directed acyclic graph, in which every node has at most one ingoing edge. The nodes of the lexicon contain words, while the edges correspond to morphological rules. The parameter of the model is the set R of morphological rules. In the following, we will define the distributions $P(R)$ and $P(L|R)$.

In order to define $P(L|R)$, we will decompose L into layers: $L_0, L_1, \ldots$ as follows: let L_0 contain all the nodes, that have no ingoing edge. Such nodes will be called *roots*.[1] Then, each L_{i+1} contains the nodes, that are derived from a node from L_i. Let rt be a probability distribution (called *root probability distribution*) over the set of all strings using letters from the target language's alphabet. For example, rt can be based on letter frequencies. Then we define:

$$P(L_0) := P(|L_0|) \cdot |L_0|! \cdot \prod_{w \in L_0} rt(w) \quad (2)$$

First, we draw the length of L_0 from some distribution. Then, each of the elements of L_0 is drawn independently from the root distribution. As the ordering of the elements is irrelevant, the result is multiplied by the number of possible orderings. Note that $P(L_0)$ does not depend on R and that rt is not a model parameter (i.e. it is fixed). We can get rid of the factorial term by using Poisson[2] distribution with parameter λ_L for $P(|L_0|)$, which yields:

$$P(L_0) := e^{-\lambda_L} \lambda_L^{|L_0|} \prod_{w \in L_0} rt(w) \quad (3)$$

[1] The notion of *root* used here has nothing to do with the definition typically used in morphology. It is meant as a root of a derivational tree, like the one shown in Fig. 1, which principally can be any word.

[2] The distribution of set length has negligible influence on the behavior of the model and is included only for formal completeness. Poisson distribution is chosen because of mathematical simplicity.

Next, we will define the probability $P(L_{i+1}|L_i, R)$ of deriving the layer L_{i+1} from L_i using the rules from set R. For each rule r, let π_r denote a probability, called *productivity* of r. Further, let $r(w)$ denote the (possibly empty) set of words resulting from applying the rule r to w.[3] Finally, let E_L denote the set of edges of L. Then:

$$P(L_{i+1}|L_i, R) := \prod_{w \in L_i} \prod_{r \in R} \prod_{w' \in r(w)} \begin{cases} \pi_r & \text{if } (w, w') \in E_L \\ 1 - \pi_r & \text{if } (w, w') \notin E_L \end{cases} \tag{4}$$

In other words, for each word from the layer L_i, each rule can apply with its inherent probability π_r. The latter corresponds to the definition of *productivity* mentioned by Aronoff [2, p. 36]. The lexicon also contains information about cases where a rule *does not* apply, the probability of which equals $1 - \pi_r$.

Finally, we can define the complete likelihood function:

$$P(L|R) := P(L_0) \cdot \prod_{i=0}^{\infty} P(L_{i+1}|L_i, R) \tag{5}$$

The product going to infinity can be justified as follows: once there is a $L_k = \emptyset$, then all further layers must also be empty and (4) yields $P(\emptyset|\emptyset, R) = 1$.

The rule set prior $P(R)$ is defined similarly to $P(L_0)$: first, we introduce a distribution $P(r)$ over single rules. For this purpose, we decompose a rule into a sequence of elementary edit operations (insertion or deletion of a character). We also introduce a third operation $COPY$ (c), which leaves an arbitrary number of characters unchanged. For example, the German rule $/Xen/_{\text{V.INF}} \rightarrow /geXt/_{\text{V.PP}}$ would be expressed as the sequence:

$$(i(\text{'g'}), i(\text{'e'}), c, d(\text{'e'}), d(\text{'n'}), i(\text{'t'}), d(\text{V.INF}), i(\text{V.PP}))$$

The distribution $P(r)$ is obtained by assigning (fixed) weights to each of the three operations and taking a probability distribution over letters and tags (e.g. according to their corpus frequency).

The next step is specifying a prior distribution for rule productivities. It is easy to check that the rule frequency in the lexicon follows a binomial distribution. Therefore, we use a standard non-informative prior $\text{BETA}(1, 1)$ for productivity. Finally, as we did with L_0, also here we use a Poisson distribution with parameter λ_R for $|R|$. Then we obtain:

$$P(R) := e^{-\lambda_R} \lambda_R^{|R|} \prod_{r \in R} P(r) P(\pi_r) \tag{6}$$

[3] In our formalism, rules are functions mapping words to *sets* of words. The set is empty if the constraints on the left-hand side of the rule are not met. Otherwise, typically a single word is produced, but cases with more than one result are also possible.

4.3 Extension with Features

The model sketched above can be further extended to include arbitrary features, like word frequency, semantic vectors, inflectional classes etc. In this case, the graph nodes contain feature vectors and the rules are conditional distributions on feature values.

As an example, consider the PoS-tag of a word. In the model presented in the previous section, it was treated like a part of the word's string representation. Instead, we can define it as a separate feature t. Then, the nodes of the lexicon are pairs (w, t), while the rules additionally contain a probability distribution (called *transformational distribution*) $\tau_r(t'|t)$ of the tag of the resulting word, given the tag of the base word. For the above-mentioned rule for German past participle formation, $\tau_r(t'|\text{V.}\textsc{Inf})$ would equal 1 for $t' = \text{V.PP}$ and 0 otherwise. For $t \neq \text{V.}\textsc{Inf}$, $\tau_r(t'|t)$ can be left undefined, because the rule does not produce any results.

Equations (3), (4) and (6) become then:

$$P(L_0) := e^{-\lambda_L} \lambda_L^{|L_0|} \prod_{(w,t) \in L_0} rt(w) P(t|w) \tag{7}$$

$$P(L_{i+1}|L_i, R) := \prod_{(w,t) \in L_i} \prod_{r \in R} \prod_{w' \in r(w)} \begin{cases} \pi_r \tau_r(t'|t) & \text{if } \exists_{t'}((w,t),(w',t')) \in E_L \\ 1 - \pi_r & \text{otherwise} \end{cases} \tag{8}$$

$$P(R) := e^{-\lambda_R} \lambda_R^{|R|} \prod_{r \in R} P(r) P(\pi_r) P(\tau_r) \tag{9}$$

In addition, we need the distribution on root tags (possibly conditioned on the string form of the word) $P(t|w)$ and the prior distribution on τ, $P(\tau)$. In this example, the former can just be based on the frequencies of the tags in training data, while the latter can be a uniform distribution on all possible tag pairs (only degenerate τ, that equal 1 for exactly one resulting tag, are taken into account).

While the above example may look overcomplicated, this formalism allows us to incorporate a large variety of features into the model. Let us consider another example: the *frequency class* of a word, defined as $f_w = \lfloor \log_2 \frac{\max_{w' \in L} freq(w')}{freq(w)} \rfloor$, where $freq$ is the corpus frequency. There are reasons to assume, that morphological rules add a roughly constant factor to the word's frequency class. Consider Fig. 2, which shows the differences in frequency class between German word pairs conforming to the rule $/X/ \rightarrow /Xs/$, in the absence of PoS-tags. The histogram forms a bell-shaped curve with mean approximately 2. Moreover, the cases near the mean correspond to regular morphological phenomena, while the tails contain mostly pairs of unrelated words, which happen to fit the pattern, like *hau*, the imperative of *hauen* ('to hit', 'to chop') and *Haus* 'house'. Assuming a Gaussian distribution of this quantity allows us to filter out much of the noise. Thus, we introduce a feature f corresponding to the frequency class of a word. Its corresponding transformational distribution $\phi_r(f'|f)$ is a Gaussian distribution with some mean μ_{ϕ_r} (being a model parameter for each rule) and unit variance. The priors $P(f|w)$ and $P(\mu_\phi)$ can be skipped at this moment for the sake of simplicity, since they have little influence on the likelihood function. Note that the frequency classes are integers, so $\phi_r(f'|f)$ is in fact an integral of the Gaussian distribution on a unit interval. The means μ_ϕ are also limited to integers so that the prior can assign non-zero probabilities to concrete values.

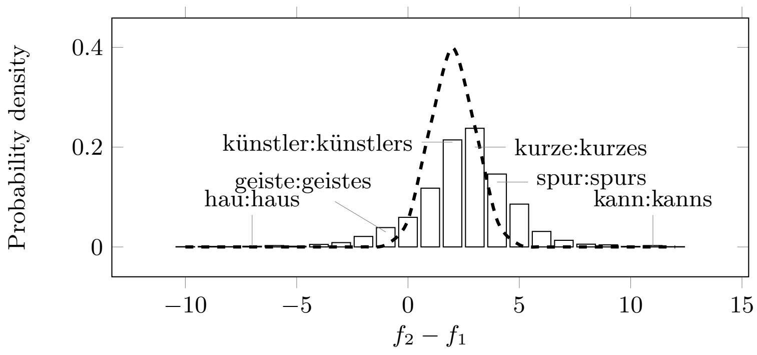

Fig. 2. Difference of frequency classes of various pairs following the rule $/X/ \rightarrow /Xs/$. The dashed line is Gaussian probability density with $\mu = 2$ and $\sigma^2 = 1$.

4.4 Local Properties

In this section, we will present some local properties of the model based on the global formulae introduced above. For the sake of simplicity, we will use the basic model without features.

Edge Score. Let's consider the contribution to the overall log-likelihood of drawing a new edge $w_1 \xrightarrow{r} w_2$, compared to a situation, where w_2 is a root. If L' denotes the lexicon with the considered edge, and L the lexicon without it, the score is given by:

$$\ln P(L'|R) - \ln P(L|R) = \ln \frac{\pi_r}{(1-\pi_r)\lambda_L rt(w_2)} \tag{10}$$

Note that this score depends on nothing else than w_2 and π_r. This fact will play an important role in the unsupervised training algorithm.

Rule Contribution. Let ν_r denote the frequency of rule r in lexicon and μ_r the number of words, to which r could be applied, but is not. The contribution of r, together with all its corresponding edges, to the overall log-likelihood is given by:

$$-\ln(\lambda_R P(r) \pi_r^{\nu_r} (1-\pi_r)^{\mu_r}) = -\ln \lambda_R - \ln P(r) - \nu_r \ln \pi_r - \mu_r \ln(1-\pi_r) \tag{11}$$

Using the above formula, we can also easily derive the optimal productivity as:

$$\pi_r^* = \frac{\nu_r}{\nu_r + \mu_r} \tag{12}$$

Word Cost. When a new word is inserted into the lexicon, it can either be attached to an existing node, or left as a root.[4] The cost of insertion is thus:

$$cost(w) = -\max(\ln rt(w), \max_{\substack{r \in R \\ L \cap r^{-1}(w) \neq \emptyset}} \ln \frac{\pi_r}{1-\pi_r}) \tag{13}$$

As the contribution to the log-likelihood is typically negative, it is plausible to call the opposite of this quantity the "cost".

Back-Formation. In some cases, adding a word as a root may seem implausible even though we do not see any possible base word. Let's consider the case of inserting *understandable*$_{\text{ADJ}}$, when *understand*$_{\text{V}}$ is not contained in the lexicon, but the rule $/X/_{\text{V}} \to /Xable/_{\text{ADJ}}$ is known and highly productive. If the root distribution *rt* has bias towards shorter words (which is the case for example for N-gram-based distributions), it may turn out, that inserting *understand* and an edge *understand* $\to$ *understandable* may yield lower cost than inserting *understandable* directly as a root. The model is thus capable of back-formation.

5 Algorithms

5.1 Preprocessing

In both unsupervised and supervised setting, the input data must be adjusted to the right format, before the actual training can start. The training data for supervised learning consist of a list of word pairs, for which the second word is known to be derived from the first. In order to convert it to a proper lexicon, a rule has to be extracted from each pair. This is done with the same algorithm as in unsupervised learning (see below).

The unsupervised learning task requires a couple more preprocessing steps. The training data consist of a list of words, with optional features (like frequency class or PoS-tag). First of all, pairs of words with sufficient string similarity are found using the FastSS algorithm [3]. The algorithm is modified in order to find morphologically related pairs: a difference of up to 5 characters at the beginning and at the end of words is allowed, as well as up to 3 characters in a single slot inside the word. While those constants can be configured arbitrarily, this setting fits to morphological rules of many languages.

Once the pairs of similar words are found, a rule is extracted from each pair. For this purpose, the Wagner-Fischer algorithm for computing string edit distance [24] is used to compute the optimal *alignment* between words. The alignment is then transformed into prefix, suffix and internal change, plus optionally PoS-tag change if tags are used.

[4] For simplicity, it is assumed here, that the newly inserted word does not overtake any child nodes from other words.

Finally, the frequency of rules is counted and only the top-N frequent rules are preserved. This filtering step has little influence on the correctness of the results and is done for performance reasons: most extracted rules are accidental similarities between unrelated words, which typically have low frequency. Filtering them out speeds up the further processing greatly. By setting N to 10000 we can be almost sure, that all real morphological rules are preserved.

5.2 Training

Problem Formulation. In the supervised learning task, we now have a full lexicon available. It remains to find a plausible rule set, which can be done in a single Maximum Likelihood estimation step. In the unsupervised setting, the graph resulting from the preprocessing steps contains all possible edges, but the subset of those, that corresponds to the lexicon structure, still has to be found. In this case, we treat the lexicon structure as a hidden variable and apply the "hard Expectation-Maximization" algorithm, as described by Samdani et al. [20], which consists of alternating Maximum Likelihood estimations of lexicon given rule set and rule set given lexicon.

Rule Optimization. Before the ML estimation steps are carried out, an additional step is performed in both unsupervised and supervised task. At the point of rule extraction, the rules were made as general as possible: only the segments that change are recorded. For example, the rule extracted from the German pair (*achten*, *achtung*) would be $/XeY/ \to /XuYg/$, although the more specific pattern $/Xen/ \to /Xung/$ is definitely more appropriate. The current step fixes this problem with the help of likelihood: for each pair of words (w_1, w_2) following a rule r, we extract *all* possible rules that describe the transformation from w_1 to w_2. Then we calculate the contribution of each rule to the overall log-likelihood using (11). Finally, we choose the set of rules that minimizes the costs.

In the above example, the original rule $r : /XeY/ \to /XuYg/$ is splitted into $r_1 : /Xen/ \to /Xung/$ and $r_2 : /XeY/ \to /XuYg/$. r_1 covers the most cases, so $\nu_{r_1} \approx \nu_r$, but $\mu_{r_1} < \mu_r$, because the constraint on the left side is stronger. Thus, the last term of (11) is weakened. Also, $\pi_{r_1} > \pi_r$, which decreases the cost further. The remaining cases, like for example the accidental similarity (*ber*, *burg*)[5], are covered by r_2. Here, $\mu_{r_2} = \mu_r + \nu_{r_1}$, but ν_{r_2} and π_{r_2} are very small. In conclusion, the following inequality is fulfilled:

$$P(r)\pi_r^{\nu_r}(1-\pi_r)^{\mu_r} < \lambda_R P(r_1)\pi_{r_1}^{\nu_{r_1}}(1-\pi_{r_1})^{\mu_{r_1}} P(r_2)\pi_{r_2}^{\nu_{r_2}}(1-\pi_{r_2})^{\mu_{r_2}} \qquad (14)$$

This justifies the splitting of r into r_1 and r_2. In unsupervised learning, this step is performed only once, before running the EM algorithm.

[5] Although *ber* is not a valid German word, it may happen to occur in the data, for example as an abbreviation or a foreign word.

Estimating Rules Given Lexicon. In this step, the productivity of each rule is set to the optimal value given by (12). Once it falls to 0, the rule is deleted. If the model uses frequency class as a feature, also the means μ_{ϕ_r} have to be estimated for each rule. This is done by setting each mean to the rounded average difference of frequency classes for the pairs of words following the rule.

Estimating Lexicon Given Rules. While searching for an optimal lexicon, we consider the edges obtained in the preprocessing steps and look for a subset of those, in which every node has at most one ingoing edge. This problem is known as *optimal branching* of a graph and can be solved with Chu-Liu-Edmonds' algorithm [21]. As weight of the edges, we use the contribution of an edge to the log-likelihood given by (10). The property, that this weight depends on nothing else than the pair of nodes between which the edge is drawn, is crucial at this point. It allows the weights to stay constant as the structure of the graph is manipulated.

Checking Rules. This additional step, performed after each iteration of the EM algorithm in unsupervised training, allows for easier elimination of "weak rules". The contribution of each rule to the log-likelihood (the "cost" of the rule), given by (11), is compared to the "gain", which is achieved by using this rule to derive words. In order to compute the gain of rule r, for each word w derived by r, we count the minimum cost of deriving w by another rule, or the cost of introducing w as a root if no other rule is possible. The sum of those costs constitutes the gain of r:

$$gain(r) := -\sum_{w:\overset{r}{\to}w} \max(\ln rt(w), \max_{\substack{r' \in R \setminus \{r\} \\ w \in rng(r')}} \ln \frac{\pi_{r'}}{1-\pi_{r'}}) \tag{15}$$

The notation $\overset{r}{\to} w$ means summing over all w that are derived by r in the present lexicon and $rng(r')$ means the set of words that can be derived by r'. Thus, words that contribute a lot to the gain of r are those, for which r has no good replacement. The rules, for which the cost exceeds the gain, are deleted.

5.3 Lexicon Search

Word Insertion. An insertion of a new word into the lexicon requires finding a position, at which the optimal cost, given by (13), is achieved. This is done by iterating over rules, in the order of decreasing productivity. For each rule r, it is assumed that the word w in consideration is derived by r. The corresponding base word w' is computed. If w' is not contained in the lexicon, back-formation is attempted, i.e. the recursively computed cost of inserting w' into the lexicon is added. The algorithm terminates if some rule r would yield bigger costs, than the previously considered rules, even in the "optimistic case", i.e. if the postulated base word was found in the lexicon. As the rules are sorted according to decreasing productivity, further search would yield even bigger costs. In this case, the best solution found so far is returned. The depth of the recursion of back-formation is typically restricted to some small number (like 1 or 2) for performance reasons.

If the model uses PoS-tags as a feature and the new word w is given without tag, this algorithm is also able to find the optimal tag. In this case, each time w is matched against a rule r, the tag standing on the right-hand side of r is used.

Lexicon Expansion. This algorithm finds morphologically motivated, but unknown words, which can be inserted into the lexicon with low cost. As the previous algorithm, the present one also considers rules in the order of decreasing productivity. For each rule r, words in lexicon are found, to which it could be applied, but is not. The results of applying r to those words are added to the list of newly generated words. Note that in this case, the cost of the newly generated word depends only on the productivity of the rule, since the base word is always contained in the lexicon. The algorithm terminates as the cost achieves some predefined threshold.

Note that the cost of adding a word may be negative ($\frac{\pi_r}{1-\pi_r} > 1$). In this case, the word is so strongly motivated, that the lexicon containing it is more likely than the one without it. Thus, even setting the cost threshold to 0 can result in generating new words.

6 Experiments

A full morphological analysis under the presented model would mean producing a graph akin to the one shown in Fig. 1, or equivalently, providing a derivation sequence for each word. This task is not yet approached. On one hand, evaluation and supervised learning would be difficult because of the lack of appropriate datasets (to our knowledge). On the other hand, first experiments with unsupervised learning produced results, that are not directly usable. For example, German prefixed verbs, like *erheben*, *beheben*, *anheben* etc. are analyzed as a "chain" of derivations (*erheben* $\rightarrow$ *beheben* $\rightarrow$ *anheben* etc.) instead of all being derived directly from a common base *heben*. This behaviour is understandable: a rule like $/erX/ \rightarrow /beX/$ has much higher productivity than $/X/ \rightarrow /beX/$, because the former applies to a more restricted set of words. Also, no property of the model punishes long chains. Such analysis might even correspond to the speaker's competence, since the knowledge, that a stem occurs with prefix *er-* makes its occurrence with prefix *be-* more likely and the whole process could also take place in the absence of *heben*. However, a method of obtaining a grammatically meaningful and practically usable analysis in the unsupervised setting still has to be found.

Nevertheless, some more specific tasks have been approached with good results, demonstrating the usefulness of the model. The following sections describe its performance in predicting unseen words, lemmatization and inflected form generation.

6.1 Lexicon Expansion

This experiment measures the capability of the model to generate new words using morphological rules. The model has been trained in the unsupervised setting on top-50k untagged wordlists obtained from the corpora of the Wortschatz Project[6] for German and Polish. All words have been lowercased and words containing characters from outside the language's alphabet were removed. Then, the lexicon expansion algorithm is used with the cost threshold of 5.0. For each language, two models were trained: with and without frequency class as a feature. The root distribution $rt(\cdot)$ is based on letter unigrams.

The precision of the results has been evaluated by matching them against lexical resources: the *list of inflected words* of the *Dictionary of the Polish Language*[7] and the German morphological analyzer Morphisto [27], respectively. An appropriate recall measure seems impossible to calculate: we would need to know *all* words, that can be predicted given the input. Instead, we plot the precision against the number of words generated.

The results are shown in Fig. 3. For both languages, the benefit of using frequency class as a feature is clearly visible, especially when generating a small number of words. At 50k words – the amount that corresponds to doubling the size of the lexicon – the precision is still around 60%. It is important to point out, that those results may be slightly lowered, because the resources used for evaluation are not perfect. In particular, the worse results on German dataset can be due to the errors of Morphisto, which fails to recognize some existing words, e.g. *Tschetschene* or *korrelieren.*

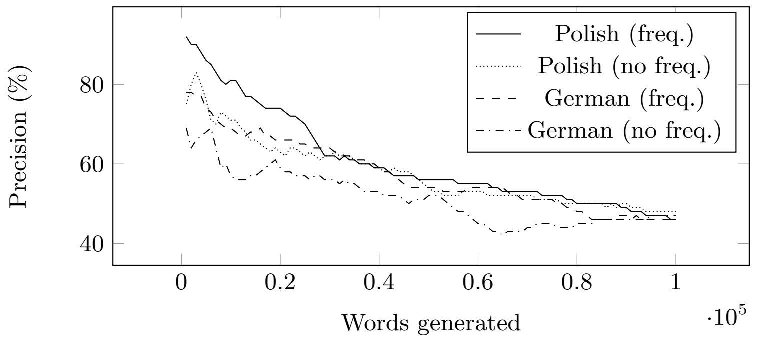

Fig. 3. Results of the unsupervised lexicon expansion task.

[6] `http://corpora.uni-leipzig.de`
[7] `http://sjp.pl/slownik/odmiany/`

6.2 Lemmatization and Tagging

The model can also be used for inflectional analysis of unknown words. In this case, the lexicon is a bipartite graph: the L_0 layer contains lemmas, and L_1 contains inflected forms. The analysis of a new word is computed by the word insertion algorithm described in Sect. 5.3. For the evaluation, we used wordlists extracted from the following tagged corpora: TIGER [4] for German and KIPI-PAN [17] for Polish. Only open lexical classes (nouns, verbs, adjectives and adverbs) were used. The only preprocessing was removing circular lemmatizations, e.g. between *Aufsichtsratsvorsitzende* and *Aufsichtsratsvorsitzender* in TIGER, which are obvious errors. In the supervised learning task, both wordlists have been divided into a training set of around 30k words and a testing set of around 10k words. Both frequency class and a full inflectional tag (containing information like case, number, gender etc. in addition to the part of speech) were used as features. The root distribution $rt(\cdot)$ is based on letter trigrams and the tag distribution $P(t|w)$ is conditioned on the last three letters of the word. Such choices account for some morphological knowledge, which improves generating unknown lemmas through back-formation.

In the *unsupervised* learning task, the input data are a list of inflected forms and a (separate) list of lemmas. The unsupervised training algorithm has been slightly modified to incorporate the knowledge of the list of lemmas: after the full graph is generated in the preprocessing step, it is filtered so that only edges connecting a lemma to an inflected form are left.

For each language and each learning task (unsupervised/supervised), four experiments have been conducted, depending on two parameters. The **Lem** parameter set to '+' means, that all necessary lemmas are known in advance, while '−' means, that only lemmas, that are contained in the training data, are known. In the latter case, the missing lemmas have to be generated using back-formation. The **Tags** parameter states whether the tag of the analyzed word is known in advance.

Baselines. The comparison of the evaluation results to other approaches is difficult, because the results depend greatly on the datasets used and the details of task formulation. Instead, we use simple ad-hoc approaches as baselines for comparison. In the unsupervised task, each inflected form is matched to the nearest lemma in terms of string edit distance. Finding the tag of the analyzed word is not attempted. For the supervised setting, a maximum-entropy classifier implemented by the `LogisticRegression` class of the Python module `scikit-learn` is trained. As features, prefixes and suffixes of length from 1 to 3 characters are used, along with the tag, provided that **Tags** parameter is set. The classifier outputs the postulated lemmatizing rule (an idea borrowed from [7]).

Results. Table 1 shows results for the unsupervised task. The three results given for each experiment correspond to the number of words, that were correctly lemmatized, correctly tagged, and both. The string edit distance-based baseline is outperformed in all experiments. Especially the discrepancy between the baseline and the model in the case, where not all lemmas are known in advance, shows, that the model is successful in generating unknown lemmas through back-formation. The performance of tag guessing is rather poor. This is not surprising, because the correct tagging often requires knowledge of the context and should be token-based, rather than type-based. However, the model could be used as a lexical predictor for a stochastic tagger.

The results for the supervised task are given in Table 2. Also here the baseline is clearly outperformed. The top-3 experiments for both languages display very good lemmatization correctness of around 90%, which means, that only the simultaneous absence of tags *and* lemmas poses a major problem for the model, while absence of only one of those is handled well. The baseline classifier on the other hand displays a sharp drop of performance in the absence of tags and cannot benefit from knowing the lemmas in advance (its results do not depend on **Lem** parameter at all).

Data			**Results**			**Baseline**		
Language	**Lem**	**Tags**	**Lem**	**Tags**	**Lem+Tags**	**Lem**	**Tags**	**Lem+Tags**
German	+	+	93%	100%	93%	84%	–	–
	+	–	80%	46%	45%	76%	–	–
	–	+	76%	100%	76%	44%	–	–
	–	–	61%	34%	28%	43%	–	–
Polish	+	+	84%	100%	84%	80%	–	–
	+	–	80%	61%	59%	67%	–	–
	–	+	80%	100%	80%	41%	–	–
	–	–	79%	61%	55%	40%	–	–

Table 1. Results for unsupervised lemmatization and tagging.

Data			**Results**			**Baseline**		
Language	**Lem**	**Tags**	**Lem**	**Tags**	**Lem+Tags**	**Lem**	**Tags**	**Lem+Tags**
German	+	+	97%	100%	97%	89%	97%	89%
	+	–	92%	38%	38%	19%	20%	19%
	–	+	90%	100%	90%	89%	97%	89%
	–	–	57%	20%	19%	19%	20%	19%
Polish	+	+	94%	100%	94%	83%	94%	83%
	+	–	93%	56%	56%	33%	36%	33%
	–	+	88%	100%	88%	83%	94%	83%
	–	–	68%	40%	38%	33%	36%	33%

Table 2. Results for supervised lemmatization and tagging.

6.3 Inflected Form Generation

Another experiment concerning inflectional morphology is generating an inflected word form given the lemma and the tag of the target form. For this purpose, the supervised models and datasets from the previous section were used. The baseline is a similar classifier as in the previous experiments: it also uses prefixes and suffixes of length from 1 to 3 as features, in addition to the lemma tag and the target tag. The resulting class is the rule that produces the inflected form. The results are given in Table 3. The baseline is slightly outperformed by our model.

Language	**Result**	**Baseline**
German	84%	83%
Polish	86%	84%

Table 3. Results for supervised inflected form generation.

7 Further work

Compounds. Conforming to the WWM theory, the model treats compounding rules as simple derivational rules, with one of the parts fixed. However, a generalization which would allow to create new compounds, in which neither part has previously been seen as a part of a compound, would be desirable. It is planned to introduce "meta-rules" (or "second-order rules") into the model: a kind of rules that would apply to *words* and produce *rules*. For example, a meta-rule $/X/ \rightarrow (/Y/ \rightarrow /XsY/)$ would apply to the word *Arbeit* to create $/Y/ \rightarrow /arbeitsY/$, which could be futher applied to *Markt* to derive *Arbeitsmarkt*. In this fashion, the analysis of compounding postulated by WWM would be maintained and the productivity of this phenomenon would be fully accounted for. However, a serious unsolved problem is that it would make the rule set depend on the lexicon, which means, that the dependency between those two would become circular.

Learning Paradigms. The notion of *paradigm* could be understood in our model as "a group of rules that frequently occur together". Those could be incorporated into the model by using a paradigm ID in the same way as a PoS tag. Knowing the paradigm IDs of words would greatly increase the productivity of rules by decreasing the μ_r-values (see (12)), because each rule could only apply to words following a certain paradigm. The paradigm IDs could thus be assigned automatically, in a fashion that maximizes the log-likelihood.

Segmentation. Although not all morphological phenomena can be described in terms of morphemes, word segmentation remains a useful description in many cases. The relational description produced by our model identifies groups of related words. Such groups can be used for segmentation into morphemes, for example with multiple sequence alignment methods explored by Kirschenbaum [12]. This additional output would make it easier to compare our model to other tools and help make it available for users, who are not convinced by the WWM theory.

Token-Based Tagging. As pointed out in Sect. 6.2, finding the right tag of an unknown word requires knowledge of the context, in addition to morphological criteria. Therefore, integrating the model with a stochastic tagger (e.g. HMM-based) will be attempted.

8 Conclusion

We have presented a probabilistic model based on the Whole Word Morphology theory. The model is an alternative to the widely used segmentation-oriented approaches. It employs a relational description of morphology, without attempting to decompose words into smaller structural units. In this way, both concatenative and non-concatenative morphological phenomena can be described in a unified way. The underlying linguistic theory is minimalistic and the behaviour of the model can be controlled by training data and prior distributions, which makes it appliable for many languages and use cases. Contrary to many machine learning approaches, the trained model, which consists of a lexicon and a set of rules with assigned probabilities, is easily interpretable and can be edited by a human expert.

The generative model contains broad knowledge, which can be attributed to "morphological competence". By manipulating its probability distributions, a single trained model can be applied to various tasks, like for instance lemmatizing unknown words, producing inflected forms, or anticipating unknown vocabulary. It is also flexible with respect to the data employed in solving those tasks: features like PoS-tag and word frequency (and possibly others) are optional, as well as labeled training data.

In addition to its machine learning capabilities, the model could perhaps also contribute to empirical linguistic studies. The definition of "productivity" as the probability of applying a rule, when the necessary conditions for applying it are met, seems reasonable from a linguistic point of view. The model accounts for phenomena like back-formation or analogy, the latter being justified in reducing the number of rules and the preference of more productive rules over less productive. As the lexicon is a part of the model and its content affects the probabilities assigned to words, the differences in morphological competence between various speakers of a language could be modeled through the differences in the content of their lexica, whereas the rules tend to be a property of the language and thus same for every speaker.

References

1. S. R. Anderson. *A-Morphous Morphology*, volume 62 of *Cambridge Studies in Linguistics*. Cambridge University Press, 1992.
2. M. Aronoff. *Word Formation in Generative Grammar*. The MIT Press, 1976.
3. T. Bocek, E. Hunt, and B. Stiller. Fast Similarity Search in Large Dictionaries. Technical report, University of Zurich, 2007.
4. S. Brants, S. Dipper, P. Eisenberg, S. Hansen, E. König, W. Lezius, C. Rohrer, G. Smith, and H. Uszkoreit. TIGER: Linguistic Interpretation of a German Corpus. *Journal of Language and Computation*, 2:597–620, 2004.
5. B. Can. *Statistical Models for Unsupervised Learning of Morphology and POS Tagging*. PhD thesis, University of York, 2011.
6. E. Chan. Learning Probabilistic Paradigms for Morphology. In *Proceedings of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology at HLT-NAACL*, pages 69–78, 2006.
7. G. Chrupała, G. Dinu, and J. van Genabith. Learning morphology with morfette. In *Proceedings of the 6th International Conference on Language Resources and Evaluation, LREC '08*, pages 2362–2367, 2008.
8. G. Durrett and J. DeNero. Supervised Learning of Complete Morphological Paradigms. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1185–1195, 2013.
9. A. Ford, R. Singh, and G. Martohardjono. *Pace Pāṇini: Towards a word-based theory of morphology*. American University Studies. Series XIII, Linguistics, Vol. 34. Peter Lang Publishing, Incorporated, 1997.
10. H. Hammarström and L. Borin. Unsupervised Learning of Morphology. *Computational Linguistics*, 37(2):309–350, 2011.
11. M. Janicki. Unsupervised Learning of A-Morphous Inflection with Graph Clustering. In *Proceedings of the Student Research Workshop associated with RANLP 2013*, pages 93–99, Hissar, Bulgaria, 2013.
12. A. Kirschenbaum. Unsupervised Segmentation for Different Types of Morphological Processes Using Multiple Sequence Alignment. In *1st International Conference on Statistical Language and Speech Processing, SLSP*, pages 152–163, Tarragona, Spain, 2013.
13. M. Kurimo, S. Virpioja, V. Turunen, and K. Lagus. Morpho Challenge 2005-2010: Evaluations and results. In *Proceedings of the 11th Meeting of the ACL-SIGMORPHON, ACL 2010*, pages 87–95, July 2010.
14. A. Mikheev. Automatic rule induction for unknown word guessing. *Computational Linguistics*, 23:405–423, 1997.
15. S. Neuvel and S. A. Fulop. Unsupervised Learning of Morphology without Morphemes. In *Proceedings of the 6th Workshop of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 31–40, 2002.
16. H. Poon, C. Cherry, and K. Toutanova. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics on NAACL 09*, pages 209–217, 2009.
17. A. Przepiórkowski. *The IPI PAN Corpus: Preliminary version*. Institute of Computer Science, Polish Academy of Sciences, Warsaw, 2004.
18. M. S. Rasooli, T. Lippincott, N. Habash, and O. Rambow. Unsupervised Morphology-Based Vocabulary Expansion. In *ACL*, pages 1349–1359, 2014.

19. T. Ruokolainen, O. Kohonen, S. Virpioja, and M. Kurimo. Supervised Morphological Segmentation in a Low-Resource Learning Setting using Conditional Random Fields. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning (CoNLL)*, pages 29–37, Sofia, Bulgaria, 2013.
20. R. Samdani, M.-W. Chang, and D. Roth. Unified Expectation Maximization. In *2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 688–698, 2012.
21. R. E. Tarjan. Finding optimum branchings. *Networks*, 7:25–35, 1977.
22. H. Tseng, D. Jurafsky, and C. Manning. Morphological features help POS tagging of unknown words across language varieties. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 32–39, 2005.
23. S. Virpioja, P. Smit, S.-A. Grönroos, and M. Kurimo. Morfessor 2.0: Python Implementation and Extensions for Morfessor Baseline. Technical report, Aalto University, Helsinki, 2013.
24. R. A. Wagner and M. J. Fischer. The String-to-String Correction Problem. *Journal of the ACM*, 21(1):168–173, 1974.
25. R. H. Wicentowski. *Modeling and Learning Multilingual Inflectional Morphology in a Minimally Supervised Framework*. PhD thesis, Johns Hopkins University, 2002.
26. D. Yarowsky and R. Wicentowski. Minimally Supervised Morphological Analysis by Multimodal Alignment. In *ACL '00*, pages 207–216, 2000.
27. A. Zielinski and C. Simon. Morphisto – An Open Source Morphological Analyzer for German. In *Finite-State Methods and Natural Language Processing, 7th International Workshop, FSMNLP 2008*, pages 224–231, Ispra, Italy, 2008.