This is an author produced version of *Probabilistic Hierarchical Clustering Of Morphological Paradigms*.

White Rose Research Online URL for this paper: http://eprints.whiterose.ac.uk/75123/

**Proceedings Paper:**
Probabilistic Hierarchical Clustering of Morphological Paradigms

Burcu Can
Department of Computer Science
University of York
Heslington, York, YO10 5GH, UK
burcucan@gmail.com

Suresh Manandhar
Department of Computer Science
University of York
Heslington, York, YO10 5GH, UK
suresh@cs.york.ac.uk

Abstract

We propose a novel method for learning morphological paradigms that are structured within a hierarchy. The hierarchical structuring of paradigms groups morphologically similar words close to each other in a tree structure. This allows detecting morphological similarities easily leading to improved morphological segmentation. Our evaluation using (Kurimo et al., 2011a; Kurimo et al., 2011b) dataset shows that our method performs competitively when compared with current state-of-art systems.

1 Introduction

Unsupervised morphological segmentation of a text involves learning rules for segmenting words into their morphemes. Morphemes are the smallest meaning bearing units of words. The learning process is fully unsupervised, using only raw text as input to the learning system. For example, the word respectively is split into morphemes respect, ive and ly. Many fields, such as machine translation, information retrieval, speech recognition etc., require morphological segmentation since new words are always created and storing all the word forms will require a massive dictionary. The task is even more complex, when morphologically complicated languages (i.e. agglutinative languages) are considered. The sparsity problem is more severe for more morphologically complex languages. Applying morphological segmentation mitigates data sparsity by tackling the issue with out-of-vocabulary (OOV) words.

In this paper, we propose a paradigmatic approach. A morphological paradigm is a pair (StemList, SuffixList) such that each concatenation of Stem+Suffix (where Stem ∈ StemList and Suffix ∈ SuffixList) is a valid word form. The learning of morphological paradigms is not novel as there has already been existing work in this area such as Goldsmith (2001), Snover et al. (2002), Monson et al. (2009), Can and Manandhar (2009) and Dreyer and Eisner (2011). However, none of these existing approaches address learning of the hierarchical structure of paradigms.

Hierarchical organisation of words help capture morphological similarities between words in a compact structure by factoring these similarities through stems, suffixes or prefixes. Our inference algorithm simultaneously infers latent variables (i.e. the morphemes) along with their hierarchical organisation. Most hierarchical clustering algorithms are single-pass, where once the hierarchical structure is built, the structure does not change further.

The paper is structured as follows: section 2 gives the related work, section 3 describes the probabilistic hierarchical clustering scheme, section 4 explains the morphological segmentation model by embedding it into the clustering scheme and describes the inference algorithm along with how the morphological segmentation is performed, section 5 presents the experiment settings along with the evaluation scores, and finally section 6 presents a discussion with a comparison with other systems that participated in Morpho Challenge 2009 and 2010.
method is similar to the Dirichlet Process (DP) based model of Goldwater et al. (2006). From this perspective, our method can be understood as adding a hierarchical structure learning layer on top of the DP based learning method proposed in Goldwater et al. (2006). Dreyer and Eisner (2011) propose an infinite Dirichlet mixture model for capturing paradigms. However, they do not address learning of hierarchy. The method proposed in Chan (2006) also learns within a hierarchical structure where Latent Dirichlet Allocation (LDA) is used to find stem-suffix matrices. However, their work is supervised, as true morphological analyses of words are provided to the system. In contrast, our proposed method is fully unsupervised.

3 Probabilistic Hierarchical Model

The hierarchical clustering proposed in this work is different from existing hierarchical clustering algorithms in two aspects:

- It is not single-pass as the hierarchical structure changes.
- It is probabilistic and is not dependent on a distance metric.

3.1 Mathematical Definition

In this paper, a hierarchical structure is a binary tree in which each internal node represents a cluster.

Let a data set be \( D = \{ x_1, x_2, \ldots, x_n \} \) and \( T \) be the entire tree, where each data point \( x_i \) is located at one of the leaf nodes (see Figure 2). Here, \( D_k \) denotes the data points in the branch \( T_k \). Each node defines a probabilistic model for words that the cluster acquires. The probabilistic model can be denoted as \( p(x_i | \theta) \) where \( \theta \) denotes the parameters of the probabilistic model.

The marginal probability of data in any node can be calculated as:

\[
p(D_k) = \int p(D_k | \theta) p(\theta | \beta) d\theta \tag{1}
\]

The likelihood of data under any subtree is defined as follows:

\[
p(D_k | T_k) = p(D_k) p(D_l | T_l) p(D_r | T_r) \tag{2}
\]

where the probability is defined in terms of left \( T_l \) and right \( T_r \) subtrees. Equation 2 provides a recursive decomposition of the likelihood in terms of the likelihood of the left and the right subtrees until the leaf nodes are reached. We use the marginal probability (Equation 1) as prior information since the marginal probability bears the probability of having the data from the left and right subtrees within a single cluster.

4 Morphological Segmentation

In our model, data points are words to be clustered and each cluster represents a paradigm. In the hierarchical structure, words will be organised in such a way that morphologically similar words will be located close to each other to be grouped in the same paradigms. Morphological similarity refers to at least one common morpheme between words. However, we do not make a distinction between morpheme types. Instead, we assume that each word is organised as a stem+suffix combination.

4.1 Model Definition

Let a dataset \( D \) consist of words to be analysed, where each word \( w_i \) has a latent variable which is
the split point that analyses the word into its stem $s_i$ and suffix $m_i$:

$$D = \{ w_1 = s_1 + m_1, \ldots, w_n = s_n + m_n \}$$

The marginal likelihood of words in the node $k$ is defined such that:

$$p(D_k) = p(S_k)p(M_k) = p(s_1, s_2, \ldots, s_n)p(m_1, m_2, \ldots, m_n)$$

The words in each cluster represents a paradigm that consists of stems and suffixes. The hierarchical model puts words sharing the same stems or suffixes close to each other in the tree. Each word is part of all the paradigms on the path from the leaf node having that word to the root. The word can share either its stem or suffix with other words in the same paradigm. Hence, a considerable number of words can be generated through this approach that may not be seen in the corpus.

We postulate that stems and suffixes are generated independently from each other. Thus, the probability of a word becomes:

$$p(w = s + m) = p(s)p(m) \quad (3)$$

We define two Dirichlet processes to generate stems and suffixes independently:

$$G_s | \beta_s, P_s \sim DP(\beta_s, P_s)$$

$$G_m | \beta_m, P_m \sim DP(\beta_m, P_m)$$

$$s | G_s \sim G_s$$

$$m | G_m \sim G_m$$

where $DP(\beta_s, P_s)$ denotes a Dirichlet process that generates stems. Here, $\beta_s$ is the concentration parameter, which determines the number of stem types generated by the Dirichlet process. The smaller the value of the concentration parameter, the less likely to generate new stem types the process is. In contrast, the larger the value of concentration parameter, the more likely it is to generate new stem types, yielding a more uniform distribution over stem types. If $\beta_s < 1$, sparse stems are supported, it yields a more skewed distribution. To support a small number of stem types in each cluster, we chose $\beta_s < 1$.

Here, $P_s$ is the base distribution. We use the base distribution as a prior probability distribution for morpheme lengths. We model morpheme lengths implicitly through the morpheme letters:

$$P_s(s_i) = \prod_{c_i \in s_i} p(c_i) \quad (4)$$

where $c_i$ denotes the letters, which are distributed uniformly. Modelling morpheme letters is a way of modelling the morpheme length since shorter morphemes are favoured in order to have fewer factors in Equation 4 (Creutz and Lagus, 2005b).

The Dirichlet process, $DP(\beta_m, P_m)$, is defined for suffixes analogously. The graphical representation of the entire model is given in Figure 3.

Once the probability distributions $G = \{G_s, G_m\}$ are drawn from both Dirichlet processes, words can be generated by drawing a stem from $G_s$ and a suffix from $G_m$. However, we do not attempt to estimate the probability distributions $G$; instead, $G$ is integrated out. The joint probability of stems is calculated by integrating out $G_s$:

$$p(s_1, s_2, \ldots, s_M) = \int p(G_s) \prod_{i=1}^{L} p(s_i|G_s) dG_s \quad (5)$$

where $L$ denotes the number of stem tokens. The joint probability distribution of stems can be tackled as a Chinese restaurant process. The Chinese restaurant process introduces dependencies between stems. Hence, the joint probability of
stems $S = \{s_1, \ldots, s_L\}$ becomes:

$$p(s_1, s_2, \ldots, s_L) = p(s_1) p(s_2 | s_1) \cdots p(s_M | s_1, \ldots, s_{M-1})$$

$$= \frac{\Gamma(\beta_s)}{\Gamma(L + \beta_s)} \beta_s^{K-1} \prod_{i=1}^{K} P_s(s_i) \prod_{i=1}^{K} (n_{s_i} - 1)!$$

(6)

where $K$ denotes the number of stem types. In the equation, the second and the third factor correspond to the case where novel stems are generated for the first time; the last factor corresponds to the case in which stems that have already been generated for $n_{s_i}$ times previously are being generated again. The first factor consists of all denominators from both cases.

The integration process is applied for probability distributions $G_m$, for suffixes analogously. Hence, the joint probability of suffixes $M = \{m_1, \ldots, m_N\}$ becomes:

$$p(m_1, m_2, \ldots, m_N) = p(m_1) p(m_2 | m_1) \cdots p(m_N | m_1, \ldots, m_{N-1})$$

$$= \frac{\Gamma(\alpha)}{\Gamma(N + \alpha)} \alpha^T \prod_{i=1}^{T} P_m(m_i) \prod_{i=1}^{T} (n_{m_i} - 1)!$$

(7)

where $T$ denotes the number of suffix types and $n_{m_i}$ is the number of stem types $m_i$ which have been already generated.

Following the joint probability distribution of stems, the conditional probability of a stem given previously generated stems can be derived as:

$$p(s_i | S^{-s_i}, \beta_s, P_s) = \begin{cases} \frac{n_{s_i}^{S^{-s_i}}}{L_1 + \beta_s} & \text{if } s_i \in S^{-s_i} \\ \frac{\beta_s P_s(s_i)}{L_1 + \beta_s} & \text{otherwise} \end{cases}$$

(8)

where $n_{s_i}^{S^{-s_i}}$ denotes the number of stem instances $s_i$ that have been previously generated, where $S^{-s_i}$ denotes the stem set excluding the new instance of the stem $s_i$.

The conditional probability of a suffix given the other suffixes that have been previously generated is defined similarly:

$$p(m_i | M^{-m_i}, \beta_m, P_m) = \begin{cases} \frac{n_{m_i}^{M^{-m_i}}}{N_1 + \beta_m} & \text{if } m_i \in M^{-m_i} \\ \frac{\beta_m P_m(m_i)}{N_1 + \beta_m} & \text{otherwise} \end{cases}$$

(9)

where $n_{m_i}^{M^{-m_i}}$ is the number of instances $m_i$ that have been generated previously where $M^{-m_i}$ is the set of suffixes, excluding the new instance of the suffix $m_i$.

A portion of a tree is given in Figure 4. As can be seen on the figure, all words are located at leaf nodes. Therefore, the root node of this subtree consists of words $\{\text{plugg+ed, skew+ed, exclaim+ed, borrow+s, borrow+ed, liken+s, liken+ed, consist+s, consist+ed}\}$.

4.2 Inference

The initial tree is constructed by randomly choosing a word from the corpus and adding this into a randomly chosen position in the tree. When constructing the initial tree, latent variables are also assigned randomly, i.e. each word is split at a random position (see Algorithm 1).

We use Metropolis Hastings algorithm (Hastings, 1970), an instance of Markov Chain Monte Carlo (MCMC) algorithms, to infer the optimal hierarchical structure along with the morphological segmentation of words (given in Algorithm 2). During each iteration $i$, a leaf node $D_i = \{w_i = s_i + m_i\}$ is drawn from the current tree structure. The drawn leaf node is removed from the tree. Next, a node $D_{k}$ is drawn uniformly from the tree.
to make it a sibling node to $D_i$. In addition to a sibling node, a split point $w_i = s_i + m_i$ is drawn uniformly. Next, the node $D_i = \{w_i = s_i + m_i\}$ is inserted as a sibling node to $D_k$. After updating all probabilities along the path to the root, the new tree structure is either accepted or rejected by applying the Metropolis-Hastings update rule. The likelihood of data under the given tree structure is used as the sampling probability.

We use a simulated annealing schedule to update $P_{\text{Acc}}$:

$$P_{\text{Acc}} = \left( \frac{p_{\text{next}}(D|T)}{p_{\text{cur}}(D|T)} \right)^{\frac{1}{\gamma}} \tag{10}$$

where $\gamma$ denotes the current temperature, $p_{\text{next}}(D|T)$ denotes the marginal likelihood of the data under the new tree structure, and $p_{\text{cur}}(D|T)$ denotes the marginal likelihood of data under the latest accepted tree structure. If $p_{\text{next}}(D|T) > p_{\text{cur}}(D|T)$ then the update is accepted (see line 9, Algorithm 2), otherwise, the tree structure is still accepted with a probability of $p_{\text{Acc}}$ (see line 14, Algorithm 2). In our experiments (see section 5) we set $\gamma$ to 2. The system temperature is reduced in each iteration of the Metropolis Hastings algorithm:

$$\gamma \leftarrow \gamma - \eta \tag{11}$$

Most tree structures are accepted in the earlier stages of the algorithm, however, as the temperature decreases only tree structures that lead to a considerable improvement in the marginal probability $p(D|T)$ are accepted.

An illustration of sampling a new tree structure is given in Figure 5 and 6. Figure 5 shows that $D_0$ will be removed from the tree in order to sample a new position on the tree, along with a new split point of the word. Once the leaf node is removed from the tree, the parent node is removed from the tree, as the parent node $D_5$ will consist of only one child. Figure 6 shows that $D_h$ is sampled to be the sibling node of $D_0$. Subsequently, the two nodes are merged within a new cluster that
introduces a new node $D_0$.

4.3 Morphological Segmentation

Once the optimal tree structure is inferred, along with the morphological segmentation of words, any novel word can be analysed. For the segmentation of novel words, the root node is used as it contains all stems and suffixes which are already extracted from the training data. Morphological segmentation is performed in two ways: segmentation at a single point and segmentation at multiple points.

4.3.1 Single Split Point

In order to find single split point for the morphological segmentation of a word, the split point yielding the maximum probability given inferred stems and suffixes is chosen to be the final analysis of the word:

$$\arg \max_j p(w_i = s_j + m_j | D_{\text{root}}, \beta_m, P_m, \beta_s, P_s)$$

where $D_{\text{root}}$ refers to the root of the entire tree.

Here, the probability of a segmentation of a given word given $D_{\text{root}}$ is calculated as given below:

$$p(w_i = s_j + m_j | D_{\text{root}}, \beta_m, P_m, \beta_s, P_s) = p(s_j | S_{\text{root}}, \beta_s, P_s) \ p(m_j | M_{\text{root}}, \beta_m, P_m)$$

where $S_{\text{root}}$ denotes all the stems in $D_{\text{root}}$ and $M_{\text{root}}$ denotes all the suffixes in $D_{\text{root}}$. Here $p(s_j | S_{\text{root}}, \beta_s, P_s)$ is calculated as given below:

$$p(s_j | S_{\text{root}}, \beta_s, P_s) = \begin{cases} n_s^{s_{\text{root}}} \over L+\beta_s & \text{if } s_j \in S_{\text{root}} \\ \beta_s P_s(s_j) \over L+\beta_s & \text{otherwise} \end{cases}$$

Similarly, $p(m_j | M_{\text{root}}, \beta_m, P_m)$ is calculated as:

$$p(m_j | M_{\text{root}}, \beta_m, P_m) = \begin{cases} n_m^{m_{\text{root}}} \over N+\beta_m & \text{if } m_j \in M_{\text{root}} \\ \beta_m P_m(m_j) \over N+\beta_m & \text{otherwise} \end{cases}$$

4.3.2 Multiple Split Points

In order to discover words with multiple split points, we propose a hierarchical segmentation where each segment is split further. The rules for generating multiple split points is given by the following context free grammar:

$$w \leftarrow s_1 \ m_1 | s_2 \ m_2$$

$$s_1 \leftarrow s \ m | s \ s$$

$$s_2 \leftarrow s$$

$$m_1 \leftarrow m \ m$$

$$m_2 \leftarrow s \ m | m \ m$$

Here, $s$ is a pre-terminal node that generates all the stems from the root node. And similarly, $m$ is a pre-terminal node that generates all the suffixes from the root node. First, using Equation 16, the word (e.g. housekeeper) is split into $s_1 \ m_1$ (e.g. housekeep+) or $s_2 \ m_2$ (house+keeper). The first segment is regarded as a stem, and the second segment is either a stem or a suffix, considering the probability of having a compound word. Equation 12 is used to decide whether the second segment is a stem or a suffix. At the second segmentation level, each segment is split once more. If the first production rule is followed in the first segmentation level, the first segment $s_1$ can be analysed as $s \ m$ (e.g. housekeep+) or $s \ s$.
(e.g. house+keep) (Equation 17). The decision to choose which production rule to apply is made using:

\[
s_1 \leftarrow \begin{cases} 
  s & \text{if } p(s|S, \beta_s, P_s) > p(m|M, \beta_m, P_m) \\ 
  m_m & \text{otherwise}
\end{cases}
\]  

(21)

where S and M denote all the stems and suffixes in the root node.

Following the same production rule, the second segment \(m_1\) can only be analysed as \(m m\) (\(e+r+\emptyset\)). We postulate that words cannot have more than two stems and suffixes always follow stems. We do not allow any prefixes, circumfixes, or infixes. Therefore, the first production rule can output two different analyses: \(s m m\) and \(s s m\) (e.g. housekeep+er and house+keep+er).

On the other hand, if the word is analysed as \(s_2 m_2\) (e.g. house+keeper), then \(s_2\) cannot be analysed further. (e.g. house). The second segment \(m_2\) can be analysed further, such that \(s m\) (stem+suffix) (e.g. keep+er, keeper+\emptyset) or \(m m\) (suffix+suffix). The decision to choose which production rule to apply is made as follows:

\[
m_2 \leftarrow \begin{cases} 
  s m & \text{if } p(s|S, \beta_s, P_s) > p(m|M, \beta_m, P_m) \\ 
  m m & \text{otherwise}
\end{cases}
\]  

(22)

Thus, the second production rule yields two different analyses: \(s s m\) and \(s m m\) (e.g. house+keep+er or house+keeper).

5 Experiments & Results

Two sets of experiments were performed for the evaluation of the model. In the first set of experiments, each word is split at single point giving a single stem and a single suffix. In the second set of experiments, potentially multiple split points are generated, by splitting each stem and suffix once more, if it is possible to do so.

Morpho Challenge (Kurimo et al., 2011b) provides a well established evaluation framework that additionally allows comparing our model in a range of languages. In both sets of experiments, the Morpho Challenge 2010 dataset is used (Kurimo et al., 2011b). Experiments are performed for English, where the dataset consists of 878,034 words. Although the dataset provides word frequencies, we have not used any frequency information. However, for training our model, we only chose words with frequency greater than 200.

In our experiments, we used dataset sizes of 10K, 16K, 22K words. However, for final evaluation, we trained our models on 22K words. We were unable to complete the experiments with larger training datasets due to memory limitations. We plan to report this in future work. Once the tree is learned by the inference algorithm, the final tree is used for the segmentation of the entire dataset. Several experiments are performed for each setting where the setting varies with the tree size and the model parameters. Model parameters are the concentration parameters \(\beta = \{\beta_s, \beta_m\}\) of the Dirichlet processes. The concentration parameters, which are set for the experiments, are 0.1, 0.2, 0.02, 0.001, 0.002.

In all experiments, the initial temperature of the system is assigned as \(\gamma = 2\) and it is reduced to the temperature \(\gamma = 0.01\) with decrements \(\eta = 0.0001\). Figure 8 shows how the log likelihoods of trees of size 16K and 22K converge in time (where the time axis refers to sampling iterations).

Since different training sets will lead to different tree structures, each experiment is repeated three times keeping the experiment setting the same.
5.1 Experiments with Single Split Points

In the first set of experiments, words are split into a single stem and suffix. During the segmentation, Equation 12 is used to determine the split position of each word. Evaluation scores are given in Table 1. The highest F-measure obtained is 51.28% with the dataset of 22K words. The scores are noticeably higher with the largest training set.

5.2 Experiments with Multiple Split Points

The evaluation scores of experiments with multiple split points are given in Table 2. The highest F-measure obtained is 62.56% with the dataset with 22K words. As for single split points, the scores are noticeably higher with the largest training set.

For both, single and multiple segmentation, the same inferred tree has been used.

5.3 Comparison with Other Systems

For all our evaluation experiments using Morpho Challenge 2010 (English and Turkish) and Morpho Challenge 2009 (English), we used 22k words for training. For each evaluation, we randomly chose 22k words for training and ran our MCMC inference procedure to learn our model. We generated 3 different models by choosing 3 different randomly generated training sets each consisting of 22k words. The results are the best results over these 3 models. We are reporting the best results out of the 3 models due to the small (22k word) datasets used. Use of larger datasets would have resulted in less variation and better results.

We compare our system with the other participant systems in Morpho Challenge 2010. Results are given in Table 6 (Virpioja et al., 2011). Since the model is evaluated using the official (hidden) Morpho Challenge 2010 evaluation dataset where we submit our system for evaluation to the organisers, the scores are different from the ones that we presented Table 1 and Table 2.

We also demonstrate experiments with Morpho Challenge 2009 English dataset. The dataset consists of 384,904 words. Our results and the results of other participant systems in Morpho Challenge 2009 are given in Table 3 (Kurimo et al., 2009). It should be noted that we only present the top systems that participated in Morpho Challenge 2009. If all the systems are considered, our system comes 5th out of 16 systems.

The problem of morphologically rich languages is not our priority within this research. Nevertheless, we provide evaluation scores on Turkish. The Turkish dataset consists of 617,298 words. We chose words with frequency greater than 50 for Turkish since the Turkish dataset is not large enough. The results for Turkish are given in Table 4. Our system comes 3rd out of 7 systems.

6 Discussion

The model can easily capture common suffixes such as -less, -s, -ed, -ment, etc. Some sample tree nodes obtained from trees are given in Table 6.
Table 4: Comparison with other unsupervised systems that participated in Morpho Challenge 2010 for Turkish.

<table>
<thead>
<tr>
<th>System</th>
<th>P(%)</th>
<th>R(%)</th>
<th>F(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morf. CatMAP</td>
<td>79.38</td>
<td>31.88</td>
<td>45.49</td>
</tr>
<tr>
<td>Aggressive Comp.</td>
<td>55.51</td>
<td>34.36</td>
<td>42.45</td>
</tr>
<tr>
<td>Prob. Clustering (multiple)</td>
<td>72.36</td>
<td>25.81</td>
<td>38.04</td>
</tr>
<tr>
<td>Iterative Comp.</td>
<td>68.69</td>
<td>21.44</td>
<td>32.68</td>
</tr>
<tr>
<td>Nicolas</td>
<td>79.02</td>
<td>19.78</td>
<td>31.64</td>
</tr>
<tr>
<td>Morf. Base.</td>
<td>89.68</td>
<td>17.78</td>
<td>29.67</td>
</tr>
<tr>
<td>Base Inference</td>
<td>72.81</td>
<td>16.11</td>
<td>26.38</td>
</tr>
</tbody>
</table>

Table 5: Sample tree nodes obtained from various trees.

```
regard+less, base+less, shame+less, bound+less, harm+less, regard+ed, relent+less
solve+d, high+priced, lower+s, lower+level, high+level, lower+income, histor+i ans
pre+mise, pre+face, pre+sumed, pre+, pre+gnant
base+ment, ail+ment, over+looked, predica+ment, deploy+ment, compart+ment, embed+i ment
anti+fraud, anti+war, anti+tank, anti+nuclear, anti+terrorism, switzer+, anti+gua, switzer+land
sharp+ened, strength+s, tight+ened, strength+ened, black+ened
inspir+e, inspir+ing, inspir+ed, inspir+es, earn+ing, ponder+ing
don grade+s, crash+ed, crash+ing, lack+ing, blind+ing, blind+, crash+, compris+ing, compris+es, sti ll+ing, compris+ed, lack+s, assist+ing, blind+ed, blind+er,
```

Sometimes similarities may not yield a valid analysis of words. For example, the prefix pre- leads the words *pre-mise, pre-sumed, pre-gnant* to be analysed wrongly, whereas *pre-* is a valid prefix for the word *pre-face*. Another nice feature about the model is that compounds are easily captured through common stems: e.g. *doubt+fire, bon+fire, gun+fire, clear+cut*.

7 Conclusion & Future Work

In this paper, we present a novel probabilistic model for unsupervised morphology learning. The model adopts a hierarchical structure in which words are organised in a tree so that morphologically similar words are located close to each other.

In hierarchical clustering, tree-cutting would be a very useful thing to do but it is not addressed in the current paper. We used just the root node as a morpheme lexicon to apply segmentation. Clearly, adding tree cutting would improve the accuracy of the segmentation and will help us identify paradigms with higher accuracy. However, the segmentation accuracy obtained without using tree cutting provides a very useful indicator to show whether this approach is promising. And experimental results show that this is indeed the case.

In the current model, we did not use any syntactic information, only words. POS tags can be utilised to group words which are both morphologically and syntactically similar.
References


