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 $\in$ StemList and Suffix $\in$ 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 .

2 Related Work

We propose a Bayesian approach for learning of paradigms in a hierarchy. If we ignore the hierarchical aspect of our learning algorithm, then our

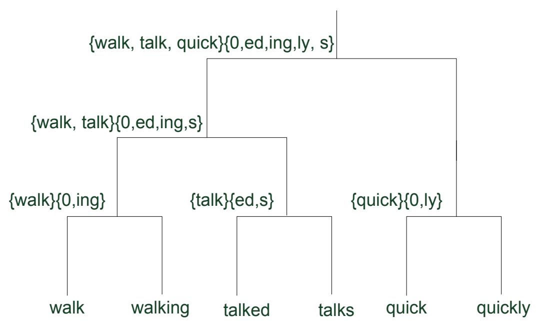

Figure 1: A sample tree structure.

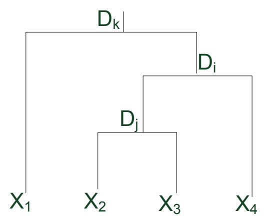

Figure 2: A segment of a tree with with internal nodes D_i, D_j, D_k having data points $\{x_1, x_2, x_3, x_4\}$. The subtree below the internal node D_i is called T_i, the subtree below the internal node D_j is T_j, and the subtree below the internal node D_k is T_k.

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 Diriclet 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 $\mathbf{D} = \{x_1, x_2, \ldots, x_n\}$ 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 θ denotes the parameters of the probabilistic model.

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

$$p(D_k) = \int p(D_k|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\beta})d\boldsymbol{\theta} \qquad (1)$$

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

$$p(D_k|T_k) = p(D_k)p(D_l|T_l)p(D_r|T_r) \qquad (2)$$

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 $\boldsymbol{D}$ 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:

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

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

$$\begin{aligned} p(D_k) &= p(S_k)p(M_k) \\ &= p(s_1, s_2, \ldots, s_n)p(m_1, m_2, \ldots, m_n) \end{aligned}$$

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:

$$p(w = s + m) = p(s)p(m) \tag{3}$$

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

$$\begin{aligned} G_s|\beta_s, P_s &\sim DP(\beta_s, P_s) \\ G_m|\beta_m, P_m &\sim DP(\beta_m, P_m) \\ s|G_s &\sim G_s \\ m|G_m &\sim G_m \end{aligned}$$

where $DP(\beta_s, P_s)$ denotes a Dirichlet process that generates stems. Here, β_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:

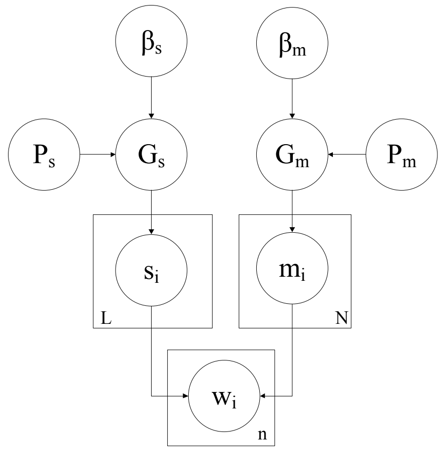

Figure 3: The plate diagram of the model, representing the generation of a word w_i from the stem s_i and the suffix m_i that are generated from Dirichlet processes. In the representation, solid-boxes denote that the process is repeated with the number given on the corner of each box.

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

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 $\mathbf{G} = \{G_s, G_m\}$ 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 $\mathbf{G}$; instead, $\mathbf{G}$ is integrated out. The joint probability of stems is calculated by integrating out G_s:

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

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:

$$
\begin{aligned}
& p(s_1, s_2, \ldots, s_L) \\
& \quad = p(s_1)p(s_2|s_1)\ldots p(s_M|s_1, \ldots, s_{M-1}) \\
& \quad = \frac{\Gamma(\beta_s)}{\Gamma(L+\beta_s)}\beta_s^{K-1}\prod_{i=1}^{K} P_s(s_i)\prod_{i=1}^{K}(n_{s_i}-1)!
\end{aligned} \tag{6}
$$

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:

$$
\begin{aligned}
& p(m_1, m_2, \ldots, m_N) \\
& \quad = p(m_1)p(m_2|m_1)\ldots p(m_N|m_1, \ldots, m_{N-1}) \\
& \quad = \frac{\Gamma(\alpha)}{\Gamma(N+\alpha)}\alpha^{T}\prod_{i=1}^{T} P_m(m_i)\prod_{i=1}^{T}(n_{m_i}-1)!
\end{aligned} \tag{7}
$$

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:

$$
\begin{aligned}
& p(s_i|S^{-s_i}, \beta_s, P_s) \\
& \quad = \begin{cases} \frac{n_{s_i}^{S^{-s_i}}}{L-1+\beta_s} & \text{if } s_i \in S^{-s_i} \\ \frac{\beta_s * P_s(s_i)}{L-1+\beta_s} & \text{otherwise} \end{cases}
\end{aligned} \tag{8}
$$

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:

$$
\begin{aligned}
& p(m_i|M^{-m_i}, \beta_m, P_m) \\
& \quad = \begin{cases} \frac{n_{m_i}^{M^{-m_i}}}{N-1+\beta_m} & \text{if } m_i \in M^{-m_i} \\ \frac{\beta_m * P_m(m_i)}{N-1+\beta_m} & \text{otherwise} \end{cases}
\end{aligned} \tag{9}
$$

where $n_{m^i}^{M_k^{-i}}$ 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.

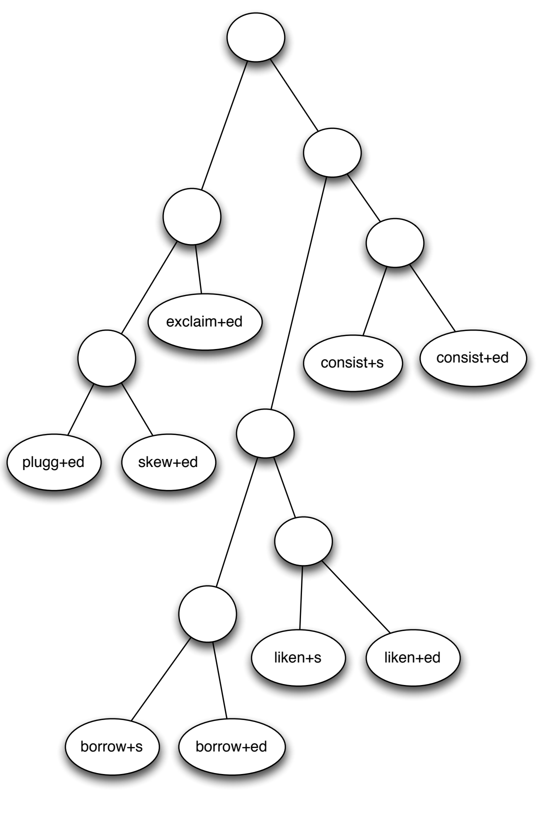

Figure 4: A portion of a sample tree.

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 {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

Algorithm 1 Creating initial tree.

```
1:  input: data $D = \{w_1 = s_1 + m_1, \ldots, w_n = s_n + m_n\}$,
2:  initialise: $root \leftarrow D_1$ where $D_1 = \{w_1 = s_1 + m_1\}$
3:  initialise: $c \leftarrow n - 1$
4:  while $c >= 1$ do
5:      Draw a word $w_j$ from the corpus.
6:      Split the word randomly such that $w_j = s_j + m_j$
7:      Create a new node $D_j$ where $D_j = \{w_j = s_j + m_j\}$
8:      Choose a sibling node $D_k$ for $D_j$
9:      Merge $D_{new} \leftarrow D_j \uplus D_k$
10:     Remove $w_j$ from the corpus
11:     $c \leftarrow c - 1$
12: end while
13: output: Initial 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_{Acc}:

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

where γ denotes the current temperature, $p_{next}(D|T)$ denotes the marginal likelihood of the data under the new tree structure, and $p_{cur}(D|T)$ denotes the marginal likelihood of data under the latest accepted tree structure. If $(p_{next}(D|T) > p_{cur}(D|T))$ then the update is accepted (see line 9, Algorithm 2), otherwise, the tree structure is still accepted with a probability of p_{Acc} (see line 14, Algorithm 2). In our experiments (see section 5) we set γ 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 lead to a considerable improvement in the marginal probability $p(D|T)$ are accepted.

Algorithm 2 Inference algorithm

```
1:  input: data $D = \{w_1 = s_1 + m_1, \ldots, w_n = s_n + m_n\}$, initial tree $T$, initial temperature of the system $\gamma$, the target temperature of the system $\kappa$, temperature decrement $\eta$
2:  initialise: $i \leftarrow 1$, $w \leftarrow w_i = s_i + m_i$, $p_{cur}(D|T) \leftarrow p(D|T)$
3:  while $\gamma > \kappa$ do
4:      Remove the leaf node $D_i$ that has the word $w_i = s_i + m_i$
5:      Draw a split point for the word such that $w_i = s_i^{'} + m_i^{'}$
6:      Draw a sibling node $D_j$
7:      $D_m \leftarrow D_i \uplus D_j$
8:      Update $p_{next}(D|T)$
9:      if $p_{next}(D|T) >= p_{cur}(D|T)$ then
10:         Accept the new tree structure
11:         $p_{cur}(D|T) \leftarrow p_{next}(D|T)$
12:     else
13:         $random \sim Normal(0, 1)$
14:         if $random < \left(\frac{p_{next}(D|T)}{p_{cur}(D|T)}\right)^{\frac{1}{\gamma}}$ then
15:             Accept the new tree structure
16:             $p_{cur}(D|T) \leftarrow p_{next}(D|T)$
17:         else
18:             Reject the new tree structure
19:             Re-insert the node $D_i$ at its previous position with the previous split point
20:         end if
21:     end if
22:     $w \leftarrow w_{i+1} = s_{i+1} + m_{i+1}$
23:     $\gamma \leftarrow \gamma - \eta$
24: end while
25: output: A tree structure where each node corresponds to a paradigm.
```

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_8 is sampled to be the sibling node of D_0. Subsequently, the two nodes are merged within a new cluster that

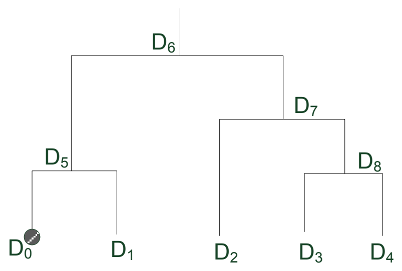

Figure 5: D_0 will be removed from the tree.

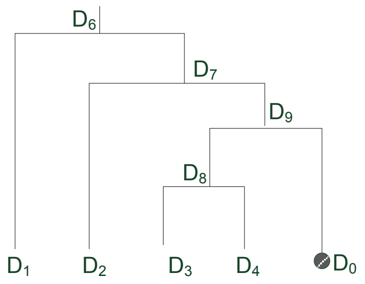

Figure 6: D_8 is sampled to be the sibling of D_0.

introduces a new node D_9.

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:

$$\arg\max_{\mathbf{j}} p(w_i = s_j + m_j | D_{root}, \beta_m, P_m, \beta_s, P_s) \tag{12}$$

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

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

$$p(w_i = s_j + m_j | D_{root}, \beta_m, P_m, \beta_s, P_s) = p(s_j | S_{root}, \beta_s, P_s)\, p(m_j | M_{root}, \beta_m, P_m) \tag{13}$$

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

$$p(s_i|S_{root}, \beta_s, P_s) = \begin{cases} \frac{n_{s_i}^{S_{root}}}{L+\beta_s} & \text{if } s_i \in S_{root} \\ \frac{\beta_s * P_s(s_i)}{L+\beta_s} & \text{otherwise} \end{cases} \tag{14}$$

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

$$p(m_i|M_{root}, \beta_m, P_m) = \begin{cases} \frac{n_{m_i}^{M_{root}}}{N+\beta_m} & \text{if } m_i \in M_{root} \\ \frac{\beta_m * P_m(m_i)}{N+\beta_m} & \text{otherwise} \end{cases} \tag{15}$$

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:

$$w \leftarrow s_1\, m_1 | s_2\, m_2 \tag{16}$$
$$s_1 \leftarrow s\, m | s\, s \tag{17}$$
$$s_2 \leftarrow s \tag{18}$$
$$m_1 \leftarrow m\, m \tag{19}$$
$$m_2 \leftarrow s\, m | m\, m \tag{20}$$

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+er) 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+$\emptyset$) or $s\, s$

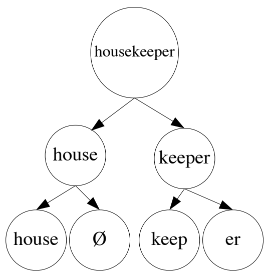

Figure 7: An example that depicts how the word *housekeeper* can be analysed further to find more split points.

(e.g. house+keep) (Equation 17). The decision to choose which production rule to apply is made using:

$$s_1 \leftarrow \begin{cases} s\ s \ \text{if}\ p(s|S,\beta_s,P_s) > p(m|M,\beta_m,P_m) \\ s\ m \ \text{otherwise} \end{cases} \tag{21}$$

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 $m\ m$ (er+$\emptyset$). 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: $s\ m\ m\ m$ and $s\ s\ m\ m$ (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:

$$m_2 \leftarrow \begin{cases} s\ m \ \text{if}\ p(s|S,\beta_s,P_s) > p(m|M,\beta_m,P_m) \\ m\ m \ \text{otherwise} \end{cases} \tag{22}$$

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

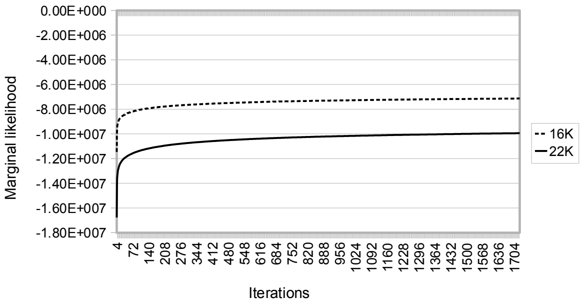

Figure 8: Marginal likelihood convergence for datasets of size 16K and 22K words.

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.

Data Size	P(%)	R(%)	F(%)	β_s, β_m
10K	81.48	33.03	47.01	0.1, 0.1
16K	86.48	35.13	50.02	0.002, 0.002
22K	89.04	36.01	51.28	0.002, 0.002

Table 1: Highest evaluation scores of single split point experiments obtained from the trees with 10K, 16K, and 22K words.

Data Size	P(%)	R(%)	F(%)	β_s, β_m
10K	62.45	57.62	59.98	0.1, 0.1
16K	67.80	57.72	62.36	0.002, 0.002
22K	68.71	62.56	62.56	0.001 0.001

Table 2: Evaluation scores of multiple split point experiments obtained from the trees with 10K, 16K, and 22K words.

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.

System	P(%)	R(%)	F(%)
Allomorf[1]	68.98	56.82	62.31
Morf. Base.[2]	74.93	49.81	59.84
PM-Union[3]	55.68	62.33	58.82
Lignos[4]	83.49	45.00	58.48
Prob. Clustering (multiple)	57.08	57.58	**57.33**
PM-mimic[3]	53.13	59.01	55.91
MorphoNet[5]	65.08	47.82	55.13
Rali-cof[6]	68.32	46.45	55.30
CanMan[7]	58.52	44.82	50.76

[1] Virpioja et al. (2009)
[2] Creutz and Lagus (2002)
[3] Monson et al. (2009)
[4] Lignos et al. (2009)
[5] Bernhard (2009)
[6] Lavallée and Langlais (2009)
[7] Can and Manandhar (2009)

Table 3: Comparison with other unsupervised systems that participated in Morpho Challenge 2009 for English.

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.

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

Table 4: Comparison with other unsupervised systems that participated in Morpho Challenge 2010 for Turkish.

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+ians
pre+mise, pre+face, pre+sumed, pre+, pre+gnant
base+ment, ail+ment, over+looked, predica+ment, deploy+ment, compart+ment, embodi+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
downgrade+s, crash+ed, crash+ing, lack+ing, blind+ing, blind+, crash+, compris+ing, compris+es, stifl+ing, compris+ed, lack+s, assist+ing, blind+ed, blind+er,

Table 5: Sample tree nodes obtained from various trees.

As seen from the table, morphologically similar words are grouped together. Morphological similarity refers to at least one common morpheme between words. For example, the words *high-priced* and *lower-level* are grouped in the same node through the word *high-level* which shares the same stem with *high-priced* and the same ending with *lower-level*.

As seen from the sample nodes, prefixes can also be identified, for example *anti+fraud, anti+war, anti+tank, anti+nuclear*. This illustrates the flexibility in the model by capturing the similarities through either stems, suffixes or prefixes. However, as mentioned above, the model does not consider any discrimination between different types of morphological forms during training. As the prefix *pre-* appears at the beginning of words, it is identified as a stem. However, identifying *pre-* as a stem does not yield a change in the morphological analysis of the word.

System	P(%)	R(%)	F(%)
Base Inference[1]	80.77	53.76	64.55
Iterative Comp.[1]	80.27	52.76	63.67
Aggressive Comp.[1]	71.45	52.31	60.40
Nicolas[2]	67.83	53.43	59.78
Prob. Clustering (multiple)	57.08	57.58	**57.33**
Morf. Baseline[3]	81.39	41.70	55.14
Prob. Clustering (single)	70.76	36.51	**48.17**
Morf. CatMAP[4]	86.84	30.03	44.63

[1] Lignos (2010)
[2] Nicolas et al. (2010)
[3] Creutz and Lagus (2002)
[4] Creutz and Lagus (2005a)

Table 6: Comparison of our model with other unsupervised systems that participated in Morpho Challenge 2010 for English.

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

Delphine Bernhard. 2009. Morphonet: Exploring the use of community structure for unsupervised morpheme analysis. In *Working Notes for the CLEF 2009 Workshop*, September.

Burcu Can and Suresh Manandhar. 2009. Clustering morphological paradigms using syntactic categories. In *Working Notes for the CLEF 2009 Workshop*, September.

Erwin Chan. 2006. Learning probabilistic paradigms for morphology in a latent class model. In *Proceedings of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology and Morphology*, SIGPHON '06, pages 69–78, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the ACL-02 workshop on Morphological and phonological learning - Volume 6*, MPL '02, pages 21–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2005a. Inducing the morphological lexicon of a natural language from unannotated text. In *In Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR 2005*, pages 106–113.

Mathias Creutz and Krista Lagus. 2005b. Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor 1.0. *Technical Report A81*.

Markus Dreyer and Jason Eisner. 2011. Discovering morphological paradigms from plain text using a dirichlet process mixture model. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 616–627, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *In Advances in Neural Information Processing Systems 18*, page 18.

W. K. Hastings. 1970. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109.

Mikko Kurimo, Sami Virpioja, Ville T. Turunen, Graeme W. Blackwood, and William Byrne. 2009. Overview and results of morpho challenge 2009. In *Proceedings of the 10th cross-language evaluation forum conference on Multilingual information access evaluation: text retrieval experiments*, CLEF'09, pages 578–597, Berlin, Heidelberg. Springer-Verlag.

Mikko Kurimo, Krista Lagus, Sami Virpioja, and Ville Turunen. 2011a. Morpho challenge 2009. `http://research.ics.tkk.fi/events/morphochallenge2009/`, June.

Mikko Kurimo, Krista Lagus, Sami Virpioja, and Ville Turunen. 2011b. Morpho challenge 2010. `http://research.ics.tkk.fi/events/morphochallenge2010/`, June.

Jean François Lavallée and Philippe Langlais. 2009. Morphological acquisition by formal analogy. In *Working Notes for the CLEF 2009 Workshop*, September.

Constantine Lignos, Erwin Chan, Mitchell P. Marcus, and Charles Yang. 2009. A rule-based unsupervised morphology learning framework. In *Working Notes for the CLEF 2009 Workshop*, September.

Constantine Lignos. 2010. Learning from unseen data. In Mikko Kurimo, Sami Virpioja, Ville Turunen, and Krista Lagus, editors, *Proceedings of the Morpho Challenge 2010 Workshop*, pages 35–38, Aalto University, Espoo, Finland.

Christian Monson, Kristy Hollingshead, and Brian Roark. 2009. Probabilistic paramor. In *Proceedings of the 10th cross-language evaluation forum conference on Multilingual information access evaluation: text retrieval experiments*, CLEF'09, September.

Lionel Nicolas, Jacques Farré, and Miguel A. Molinero. 2010. Unsupervised learning of concatenative morphology based on frequency-related form occurrence. In Mikko Kurimo, Sami Virpioja, Ville Turunen, and Krista Lagus, editors, *Proceedings of the Morpho Challenge 2010 Workshop*, pages 39–43, Aalto University, Espoo, Finland.

Matthew G. Snover, Gaja E. Jarosz, and Michael R. Brent. 2002. Unsupervised learning of morphology using a novel directed search algorithm: Taking the first step. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 11–20, Morristown, NJ, USA. ACL.

Sami Virpioja, Oskar Kohonen, and Krista Lagus. 2009. Unsupervised morpheme discovery with allomorfessor. In *Working Notes for the CLEF 2009 Workshop*. September.

Sami Virpioja, Ville T. Turunen, Sebastian Spiegler, Oskar Kohonen, and Mikko Kurimo. 2011. Empirical comparison of evaluation methods for unsupervised learning of morphology. In *Traitement Automatique des Langues*.