MorphoNet: Exploring the Use of Community Structure for Unsupervised Morpheme Analysis

Delphine Bernhard
Ubiquitous Knowledge Processing Lab
Computer Science Department
Technische Universität Darmstadt, Germany
`http://www.ukp.tu-darmstadt.de`

Abstract

This paper investigates a novel approach to unsupervised morphology induction relying on community detection in networks. In a first step, morphological transformation rules are automatically acquired based on graphical similarities between words. These rules encode substring substitutions for transforming one word form into another. The transformation rules are then applied to the construction of a lexical network. The nodes of the network stand for words while edges represent transformation rules. In the next step, a clustering algorithm is applied to the network to detect families of morphologically related words. Finally, morpheme analyses are produced based on the transformation rules and the word families obtained after clustering. While still in its preliminary development stages, this method obtains encouraging results at Morpho Challenge 2009, which demonstrate the viability of the approach.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: I.2.6 Learning; I.2.7 Natural Language Processing

General Terms

Algorithms, Experimentation, Languages

Keywords

morphology induction, unsupervised learning, network analysis

1 Introduction

Unsupervised morphology induction, which is the goal of the Morpho Challenge competition series, consists in automatically discovering a word's morphemes using only minimal resources such as a list of the words in the target language and a text corpus. Ideally, unsupervised algorithms should be able to learn the morphology of a large variety of languages; for Morpho Challenge 2009, the target languages were English, Finnish, German, Turkish and Arabic.

For our participation at Morpho Challenge 2009 we developed a novel method for unsupervised morphology induction called *MorphoNet*. MorphoNet relies on a *network* representation of morphological relations between words, where nodes correspond to whole word forms and edges encode morphological relatedness. Networks have been successfully used in recent years to represent linguistic phenomena for tasks such as word clustering [16], word sense disambiguation [17], summarisation, or keyword extraction [18]. Moreover, network-based methods have been shown to perform well for a wide range of NLP applications. In line with this body of research, we propose to represent morphological phenonema as a network. This approach has two major advantages. First, it is theoretically grounded in linguistic theories such as the Network Model by J. Bybee [6] or whole word morphology [19]. It differs from traditional linear concatenative approaches to morphology in that words, and not morphemes, constitute the basic unit of analysis. Second, it enables the use of effective network-based clustering and ranking methods. Our model thus benefits from research done on graphs in other domains such as sociology [21] or other areas of NLP. We especially investigate the use of *community structure* for morphology induction. Networks with community structure contain groups of nodes with dense interconnections; in our case, communities correspond to families of morphologically related words. Communities can be automatically identified in networks with community detection algorithms. To our knowledge, this is the first time that community detection algorithms are applied to the task of unsupervised morphology induction.

Though in its very early development stages, the approach yields promising results at Morpho Challenge 2009 when compared to standard baselines such as the Morfessor algorithms [7, 8].

The article is structured as follows. In the next section, we report related work. Next, we describe our method for building lexical networks. In Section 4, we explain how word families can be discovered based on the network structure, while in Section 5 we detail our approach for obtaining morpheme analyses. Evaluation results are given in Section 6.

2 Related Work on Morphology Induction

Morphological analysis is useful for many applications like speech recognition and synthesis, automatic translation or information retrieval. However, all these applications of morphology necessitate morphological resources which are not available for all languages, or, when available, are often incomplete. Much research has therefore been devoted to the unsupervised acquisition of morphological knowledge.

Methods for the unsupervised acquisition of morphological knowledge can be classified according to the intended result: (i) identification of morphologically related words (*clustering*), (ii) splitting of words into morphs (*segmentation*), and (iii) identification of morphemes (*analysis*). Morpheme analysis is the goal of the latest Morpho Challenge competitions, while for some applications, such as information retrieval, it is often sufficient to retrieve morphologically related words without proceeding to a full analysis. The identification of morphologically related words has been attempted by unsupervised methods [5] as well as approaches using dictionaries as input data [15].

Segmentation is certainly the method which has gathered the largest amount of interest in the NLP research community [4, 7, 9, 10]. It follows linear concatenative approaches to morphology such as item-and-arrangement, which postulates that words are formed by putting morphemes together. There are, however, some well known limitations to purely concatenative approaches, which are seldom dealt with by unsupervised segmentation methods. These phenomena include: (a) Ablaut and umlaut, i.e. vowel changes within a base as in English *sing*, *sang*, *sung* or German *Kloster* (singular) and *Klöster* (plural); (b) Infixation, i.e. affixes which are found within a base; (c) Expletive infixation, such as *-bloody-* in *absobloodylutely*; (d) Root-and-pattern morphology, as in Arabic.[1] In order to address these limitations, our method makes no assumption on the internal structure and morphotactics of words. It identifies flexible word transformation rules which encode substring substitutions for transforming one word form into another. These transformation rules are not limited to concatenative processes such as prefixation or suffixation (see Section 3.2).

[1]We refer the reader to Aronoff and Fudeman [1] and Bauer [3] for clear and short introductions to the listed phenomena.

Unsupervised methods rely on many properties for morphology induction, which are too numerous to be listed here. The most obvious cue is usually *graphical relatedness*: two words which share a long enough common substring are likely to be morphologically related. Graphical relatedness can be estimated by measures of orthographic distance [2] or by finding the longest initial (or final) substring [13, 22]. Our system is related to these methods in that it uses fuzzy string similarity to bootstrap the morphology induction process.

3 Lexical Networks

3.1 Use of Graphs for Morphology Induction

A network can be mathematically represented as a graph. Formally, a graph G is a pair (V, E), where V is a set of vertices (nodes) and $E \subseteq V \times V$ is a set of edges (lines, links). The main advantage of graphs is that they make it possible to take into account multiple dependencies across elements, so that the whole network plays an important role on the results obtained for a single element.

The lexical networks built by our method consist of word nodes linked by edges which encode morphological relations. Similar lexical networks have been previously described by Hathout [14]. Our approach differs however from Hathout's in two main aspects: (i) it is fully unsupervised and uses only a raw list of words as input, while Hathout's method acquires suffixation patterns from WordNet, and (ii) we attempt to take a broader range of morphological phenomena into account by acquiring morphological transformation rules which are not limited to suffixation.

3.2 Acquisition of Morphological Transformation Rules

The first step in our method consists in acquiring a set of *morphological transformation rules*. Morphological transformation rules make it possible to transform one word into another by performing substring substitutions. We represent a rule R with the following notation: `pattern` $\rightarrow$ `repl`, where `pattern` is a regular expression and `repl` is the replacement with backreferences to capturing groups in the pattern. For instance, the rule `^(.+)ly$` $\rightarrow$ `\1` applies to the word *totally* to produce the word *total*.

Transformation rules are related to, though more general than, the notion of affix pairs used in many methods for the unsupervised acquisition of morphological knowledge, under different names: patterns [14], rules [22], or transforms [12].

The main advantage of transformation rules over prefix or suffix pairs is that they are not limited to concatenative processes, which is especially useful for languages such as Arabic, e.g. when inducing rules for word pairs such as *kataba* (he wrote) and *kutiba* (it was written).

These rules are acquired using a subset L of the wordlist W provided for each language. In our experiments, we used the 10,000 most frequent words whose length exceeds the average word (type) length.[2] The method used to acquire the rules is described in detail in Algorithm 1.

For each word w in the list L we retrieve graphically similar words (Line 5, get_close_matches) using a *gestalt* approach to fuzzy pattern matching based on the Ratcliff-Obershelp algorithm.[3] For example, given the target word *democratic*, the following close matches are obtained: *undemocratic*, *democratically*, *democrats*, *democrat's*, *anti-democratic*. We then obtain rules (Line 7, get_rule_from_word_pair) by comparing the target word with all its close matches and identifying the matching subsequences;[4] for instance given the word *democratic* and its close match *undemocratic*, we obtain the following rule: `^un(.+)$` $\rightarrow$ `\1`.

We have kept all rules which occur at least twice in the training data.[5] Moreover, no attempt is made to distinguish between inflection and derivation.

[2]Except for Arabic, where there are only 9,641 word forms which are longer than the average word length in the vowelized version and 6,707 in the non-vowelized version.

[3]We used the implementation provided by the Python *difflib* module with the cutoff argument set to 0.8.

[4]Matching subsequences are identified by the get_matching_blocks Python method.

[5]For Arabic, we even kept all rules given the small size of the input word list.

Algorithm 1 Procedure for the acquisition of morphological transformation rules, given an input list of words L.

```
1: $rules \leftarrow \emptyset$
2: $n \leftarrow \mathsf{len}(L)$
3: for $i = 1$ to $n$ do
4:     $w \leftarrow L[i]$
5:     $matches \leftarrow \mathsf{get\_close\_matches}(w, L[i+1:n])$
6:     for $w_2$ in $matches$ do
7:         $r \leftarrow \mathsf{get\_rule\_from\_word\_pair}(w, w_2)$
8:         add $r$ to $rules$
9:     end for
10: end for
11: return $rules$
```

Table 1 lists the number of transformation rules obtained from the datasets provided for Morpho Challenge 2009[6] along with some examples:

Language	Word list	# rules	Example
English	wordlist.eng	834	`^re(.+)s$` → `\1`
Finnish	wordlist.fin	1,472	`^(.+)et$` → `\1ia`
German	wordlist.ger	771	`^(.+)ungen$` → `\1t`
Turkish	wordlist.tur	3,494	`^y(.+)z(.+)$` → `\1C\2`
Arabic vowelized	wordlist.vowara	8,974	`^(.+)iy(.+)$` → `\1uw\2`
Arabic non-vowelized	wordlist.nvara	2,174	`^b(.+)$` → `\1`

Table 1: Morphological transformation rules acquired for the input datasets.

3.3 Construction of a Lexical Network

Once transformation rules have been acquired, they are used to build a lexical network represented as a graph. Nodes in the graph represent words from the input word list W. Two words w_1 and w_2 are connected by an edge if there exists a transformation rule R such that $\mathrm{R}(w_1) = w_2$. The graph obtained using this method is directed based on the direction of the rules applied. Figure 1 displays an example lexical network.

4 Acquisition of Word Families

The graphs we obtain usually contain one large connected component, along with smaller connected components. Extracting connected components is thus not reliable enough to identify word families, i.e. groups of words which are related both semantically and orthographically. For instance, the lexical network depicted in Figure 1 contains one large connected component, which clearly consists of two different word families. The induction of word families can be formulated as a classical problem of community detection in graphs, and thus be solved by clustering algorithms.

Communities correspond to groups of tightly-knit nodes characterised by a high intra-group density and a lower inter-group density [20]. There are several methods to detect communities in graphs. Markov Clustering [23] for instance consists in partitioning a graph by simulating random walks in the graph; it has been used to detect communities in a graph of nouns by Dorow et al. [11]. The community detection method described by Newman [20] has been applied to natural language data by Matsuo et al. [16] for graphs based on word similarity measures by web counts.

[6]`http://www.cis.hut.fi/morphochallenge2009/`

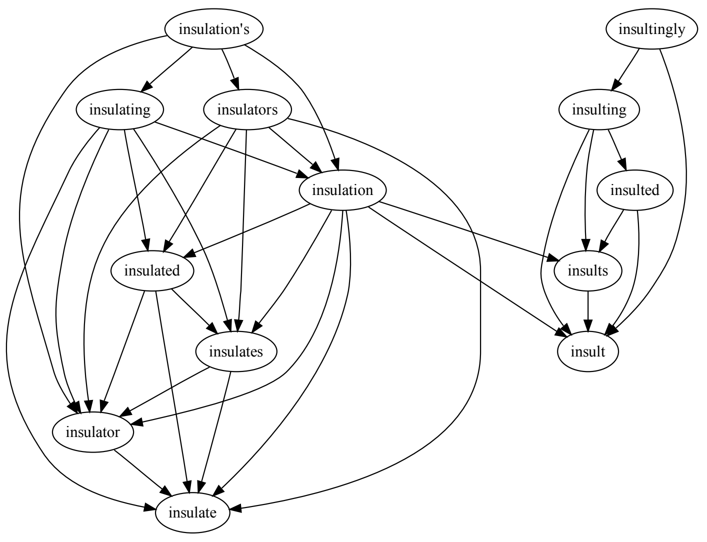

Figure 1: Example lexical network.

The method proposed by Newman relies on the modularity function Q which measures the quality of a division of a graph into communities. The advantages of this method are that it is not necessary to know the number of communities beforehand and it needs no fine parameter tuning. Modularity compares the number of edges within communities to the number of expected edges:

$$Q = \sum_i (e_{ii} - (\sum_j e_{ij})^2)$$

where e_{ii} is the fraction of the edges in the network that connect nodes within community i, e_{ij} is one-half of the fraction of edges in the network that connect nodes in community i to those in community j and $\sum_j e_{ij}$ is the fraction of edges connected to nodes in community i.

A good division corresponds to more edges within communities than would be expected by random chance, that is to say a positive modularity value Q. Modularity is high when there are many edges within communities and few between them. Figure 2 illustrates the results of Newman's algorithm on the lexical network of Figure 1: in this case, two communities are identified.

The main difficulty lies in finding the division which yields the best value for Q. It is of course infeasible to test each possible division of the network. Newman [20] therefore proposes a method of agglomerative hierarchical clustering starting from communities made of a single node. Communities are repeatedly joined together in pairs, choosing the join that leads to the biggest increase (or slightest decrease) of Q. The best partition of the network in communities corresponds to the biggest value of Q.

Our experiments with the Newman Clustering algorithm have nevertheless shown that it tends to detect bigger communities than wanted, thus decreasing the precision. We have therefore added an additional constraint on possible joins by measuring the density of edges across communities (*cross-community edge density*).

Cross-community edge density between communities A and B is defined as follows:

$$D_{AB} = \frac{\mathsf{number\_of\_edges}(A, B)}{|A| \times |B|}$$

where $\mathsf{number\_of\_edges}(A, B)$ is the number of edges linking nodes in community A to nodes in community B, and $|A|$ and $|B|$ are the number of nodes in community A and B, respectively.

The minimum cross-community edge density is fixed by a parameter d whose value ranges from 0 to 1.

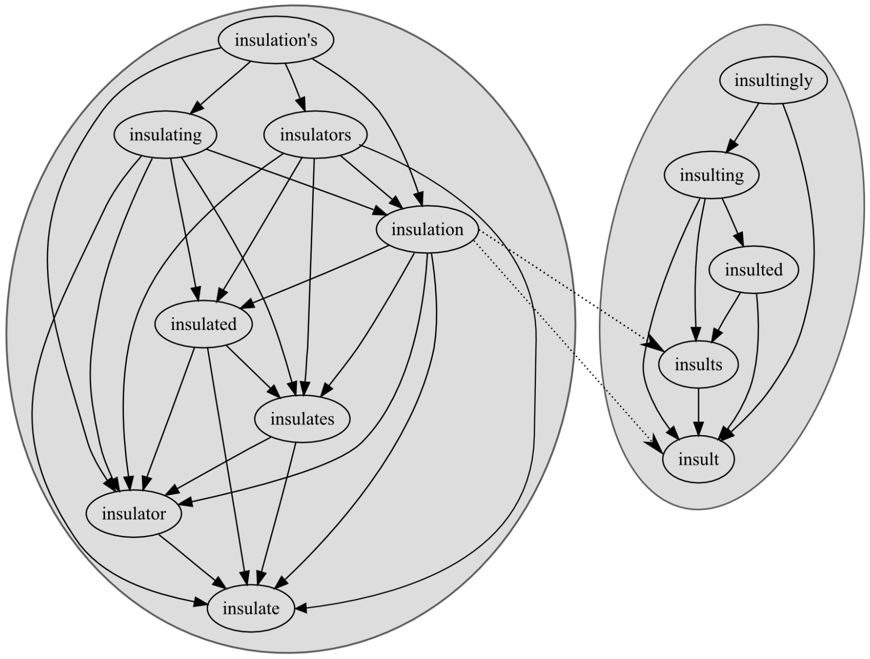

Figure 2: Illustration of Newman Clustering on a lexical network: two communities have been detected.

Table 2 displays several word families identified with this clustering algorithm. Most families are very precise, but expected families are fragmented into several smaller families, which lowers recall.

5 Morpheme Analyses

After performing clustering, morpheme analyses are obtained based on the word families identified and the transformation rule edges linking words which belong to the same family. First, a representative word is identified for each word family: this is the shortest word in the family; in case of a tie, the most frequent among the shortest words is chosen. The full morpheme analysis for a word form w consists of its family representative and a string representation of the transformation rules that apply to w. The method is detailed in Algorithm 2.

Algorithm 2 Procedure for obtaining the morpheme analyses, given a word family C and the lexical network G.

```
1: $analyses[*] \leftarrow \emptyset$
2: $subg \leftarrow \mathsf{get\_subgraph}(G, C)$
3: for edge $(w_1, w_2, rule)$ in $subg$ do
4:     $analyses[w_1] \leftarrow analyses[w_1] \cup \mathsf{to\_plain\_string}(rule.pattern)$
5:     $analyses[w_2] \leftarrow analyses[w_2] \cup \mathsf{to\_plain\_string}(rule.repl)$
6: end for
7: $rep \leftarrow \mathsf{get\_family\_representative}(C)$
8: for word $w$ in word family $C$ do
9:     $analyses[w] \leftarrow analyses[w] \cup rep$
10: end for
11: return $analyses$
```

1	absorbers absorbing absorber absorbes re-absorbing absorbingly absorb absorbs reabsorb // absorbable non-absorbable absorbables // super-absorbent super-absorbency superabsorbent // aborbed unabsorbed self-absorbed absorbed well-absorbed non-absorbed // high-absorbency absorbent absorbant absorbency absorbents absorbencies
2	well-documented undocumented documented document's // documents documents' documentation documention document-based documenting document // documentarian documentary's documentary documentaries
3	friendship's friendship friendships // friend's friendly's fiends friendy friends friendlier friendly firends fiend friends' friend unfriendly friendlies
4	emigrants emigrants' emigrant emigrant's // emigrators emgrated emigrates emigrated emigrations emigration emigratiion emigrate emigrating // migrated outmigration outmigration migration migrates migratory migrations transmigration non-migratory migrating
5	sparkler sparkling sparkles sparkled sparklers non-sparkling sparkle sparklingly

Table 2: Example word families obtained for English. The sign "//" marks a community boundary indicating family separations. The parameter d was set to 0.1.

Example Consider for instance the communities represented in Figure 2. The representative for the word family {*insulted*;*insulting*;*insult*;*insults*;*insultingly*} is *insult* since it is the shortest word. The complete analyses for the words are the following:

```
insultingly   insult _ly _ingly
insulting     insult _ing
insulted      insult _ed
insults       insult _s
insult        insult
```

Two transformation rules apply to the word *insultingly*: `^(.+)ly$` → `\1` and `^(.+)ingly$` → `\1`, which are represented in the final analysis as `_ly _ingly`.

6 Evaluation

In this section, we report the results obtained by MorphoNet at Morpho Challenge 2009 competitions 1 (linguistic evaluation) and 2 (information retrieval). For all languages, the value of parameter d (cross-community edge density) was empirically set to 0.1 for community detection.

6.1 Morpho Challenge Competition 1

Tables 3 to 8 contain the results of the linguistic evaluation (competition 1), including both the results for sample gold standards (dev) and for the final evaluation dataset (all). We also list the results obtained by the reference methods provided by the organisers: Morfessor baseline (Morf. baseline) and Morfessor CatMAP (Morf. CatMAP). Results are measured in terms of Precision (P), Recall (R) and F-Measure (F).

Method	**P**	**R**	**F**
MorphoNet - dev	68.00%	45.65%	54.63%
MorphoNet - all	65.08%	47.82%	55.13%
Morf. baseline	74.93%	49.81%	59.84%
Morf. CatMAP	84.75%	35.97%	50.50%

Table 3: English results.

Method	**P**	**R**	**F**
MorphoNet - dev	58.80%	28.98%	38.83%
MorphoNet - all	67.41%	30.19%	41.71%
Morf. baseline	81.70%	22.98%	35.87%
Morf. CatMAP	71.08%	38.92%	50.30%

Table 4: German results.

Method	P	R	F
MorphoNet - dev	58.84%	22.18%	32.21%
MorphoNet - all	63.35%	22.62%	33.34%
Morf. baseline	89.41%	15.73%	26.75%
Morf. CatMAP	79.01%	31.08%	44.61%

Table 5: Finnish results.

Method	P	R	F
MorphoNet - dev	59.92%	33.30%	42.81%
MorphoNet - all	61.75%	30.90%	41.19%
Morf. baseline	89.68%	17.78%	29.67%
Morf. CatMAP	79.38%	31.88%	45.49%

Table 7: Turkish results.

Method	P	R	F
MorphoNet - dev	90.90%	3.83%	7.36%
MorphoNet - all	92.52%	2.91%	5.65%
Morf. baseline	86.87%	4.90%	9.28%
Morf. CatMAP	-	-	-

Table 6: Arabic vowelized results.

Method	P	R	F
MorphoNet - dev	86.51%	5.04%	9.53%
MorphoNet - all	90.49%	4.95%	9.39%
Morf. baseline	91.77%	6.44%	12.03 %
Morf. CatMAP	-	-	-

Table 8: Arabic non-vowelized results.

Except for Arabic, where no results have been provided for Morfessor CatMAP, MorphoNet obtains intermediate results between both Morfessor systems. In Finnish, German and Turkish, MorphoNet performs better than Morfessor baseline, but worse than Morfessor CatMAP. In English, MorphoNet performs better than Morfessor CatMAP and worse than Morfessor baseline.

The results show that MorphoNet consistently obtains better precision than recall, especially in Arabic. The method relies on a list of transformation rules which are automatically acquired. It is therefore likely that some important rules are missing, leading to low recall. This problem might be solved by performing multiple iterations of rule induction and clustering or by applying rules in a cascaded manner, so that one rule applies to the output of another rule.

Moreover, the procedure for obtaining morpheme analyses is still very coarse and could easily be improved by detecting composite morphemes. For instance, `_ingly` could be decomposed into `_ing` and `_ly`.

Finally, transformation rules could be weighted by their frequency to improve clustering.

6.2 Morpho Challenge Competition 2

Table 9 summarises the results of the information retrieval (IR) task (competition 2). Results for the reference systems are also provided, as well as results without morpheme analysis (no analysis). The best results are in bold.

Method	English	German	Finnish
MorphoNet	0.3560	0.3167	0.3668
Morfessor baseline	0.3861	**0.4656**	0.4425
Morfessor CatMAP	0.3713	0.4642	0.4441
Snowball	**0.4081**	0.3865	0.4275
TWOL - first	0.3957	-	**0.4976**
TWOL - all	0.3922	-	0.4845
Grammatical - first	0.3734	0.3353	0.4312
Grammatical - all	0.3542	0.3014	0.4090
No analysis	0.3293	0.3509	0.3519

Table 9: IR results (average precision AP).

MorphoNet improves the IR results over unanalysed words for English and Finnish, but not for German. While it is difficult to come up with a clear explanation, this might be due to the compounding nature of German. Indeed, the MorphoNet system does not directly cope with compounding for the time being, which might be detrimental to the IR task.

7 Conclusions and Future Work

We have described a novel linguistically motivated approach to unsupervised morpheme analysis relying on a network representation of morphological relations between words. Due to the underlying network representation, it is possible to use community detection and ranking methods devised for other kinds of data. This approach is still in its very early stage, yet the results obtained at Morpho Challenge 2009 demonstrate that it yields very promising results and thus deserves further investigation.

The method described in this paper can be considered as a baseline for network-based morphology induction. It leaves lots of room for improvement. A first objective would be to obtain a better recall for morpheme analysis. This necessitates to provide a better mechanism for the acquisition of transformation rules. It should be possible to perform multiple iterations of the rule induction and clustering cycle or to apply rules in a cascaded manner. This is especially needed for languages which are morphologically more complex than English such as Turkish or Finnish. Also, we have not weighted the edges in the graph, which could be useful to improve clustering.

The clustering method performs hard-clustering: each word belongs to only one family. This is especially detrimental for languages like German, for which it would be desirable to allow multiple family membership in order to take compounding into account. In the future, we would therefore like to better address compounding.

Graphs also open up the way for a new form of modelisation of morphology enabling the analysis of crucial morphological properties. Node properties in the graph could be used to rank nodes and detect base words in families, using algorithms such as PageRank. Moreover, edge properties could be employed to differentiate between different forms of morphological processes such as inflection and derivation. We will consider these posibilities in our future work.

References

[1] Mark Aronoff and Kirsten Anne Fudeman. *What is morphology?* Wiley-Blackwell, 2005.

[2] Marco Baroni, Johannes Matiasek, and Harald Trost. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In *Proceedings of the ACL Workshop on Morphological and Phonological Learning 2002*, pages 48–57, 2002.

[3] Laurie Bauer. *Introducing Linguistic Morphology.* Georgetown University Press, 2003. 2nd edition.

[4] Delphine Bernhard. Unsupervised Morphological Segmentation Based on Segment Predictability and Word Segments Alignment. In *Proceedings of the Pascal Challenges Workshop on the Unsupervised Segmentation of Words into Morphemes*, pages 19–23, April 2006.

[5] Delphine Bernhard. Apprentissage non supervisé de familles morphologiques par classification ascendante hiérarchique. In *Actes de la 14e conférence sur le Traitement Automatique des Langues Naturelles – TALN 2007*, pages 367–376, 2007.

[6] Joan Bybee. *Morphology: A Study of the Relation between Meaning and Form.* Benjamins, Philadelphia, 1985.

[7] Mathias Creutz and Krista Lagus. Unsupervised Discovery of Morphemes. In *Proceedings of the ACL Workshop on Morphological and Phonological Learning 2002*, pages 21–30, 2002.

[8] Mathias Creutz and Krista Lagus. Inducing the Morphological Lexicon of a Natural Language from Unannotated Text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05)*, 2005.

[9] Sajib Dasgupta and Vincent Ng. High-Performance, Language-Independent Morphological Segmentation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2007)*, pages 155–163, 2007.

[10] Vera Demberg. A Language-Independent Unsupervised Model for Morphological Segmentation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 920–927, 2007.

[11] Beate Dorow, Dominic Widdows, Katerina Ling, Jean-Pierre Eckmann, Danilo Sergi, and Elisha Moses. Using Curvature and Markov Clustering in Graphs for Lexical Acquisition and Word Sense Discrimination. In *2nd MEANING Workshop*, 2005.

[12] Dayne Freitag. Morphology Induction from Term Clusters. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 128–135, 2005.

[13] Eric Gaussier. Unsupervised learning of derivational morphology from inflectional lexicons. In *Proceedings of the Workshop on Unsupervised Methods in Natural Language Processing*, 1999.

[14] Nabil Hathout. From WordNet to CELEX: acquiring morphological links from dictionaries of synonyms. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, pages 1478–1484, 2002.

[15] Nabil Hathout. Acquistion of the Morphological Structure of the Lexicon Based on Lexical Similarity and Formal Analogy. In *Proceedings of the 3rd Textgraphs workshop on Graph-based Algorithms for Natural Language Processing (COLING 2008)*, pages 1–8, 2008.

[16] Yutaka Matsuo, Takeshi Sakaki, Kôki Uchiyama, and Mitsuru Ishizuka. Graph-based Word Clustering using a Web Search Engine. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 542–550, 2006.

[17] Rada Mihalcea. Unsupervised Large-Vocabulary Word Sense Disambiguation with Graph-based Algorithms for Sequence Data Labeling. In *Proceedings of the HLT/EMNLP 2005 Conference*, pages 411–418, 2005.

[18] Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Texts. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 404–411, 2004.

[19] Sylvain Neuvel and Sean A. Fulop. Unsupervised Learning of Morphology Without Morphemes. In *Proceedings of the ACL Workshop on Morphological and Phonological Learning 2002*, pages 31–40, 2002.

[20] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69, 2004.

[21] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69, 2004.

[22] Patrick Schone and Daniel Jurafsky. Knowledge-Free Induction of Morphology Using Latent Semantic Analysis. In *Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop*, pages 67–72, 2000.

[23] Stijn van Dongen. *Graph Clustering by Flow Simulation.* PhD thesis, University of Utrecht, 2000.