Little by Little: Semi Supervised Stemming through Stem Set Minimization

Vasudevan N, Pushpak Bhattacharyya
Dept. of Computer Science and Engg.
IIT Bombay, Mumbai
{vasudevan,pb}@cse.iitb.ac.in

Abstract

In this paper we take an important step towards completely unsupervised stemming by giving a scheme for semi supervised stemming. The input to the system is a list of word forms and suffixes. The motivation of the work comes from the need to create a root or stem identifier for a language that has electronic corpora and some elementary linguistic work in the form of, say, suffix list. The scope of our work is suffix based morphology, (*i.e.*, no prefix or infix morphology). We give two greedy algorithms for stemming. We have performed extensive experimentation with four languages: English, Hindi, Malayalam and Marathi. Accuracy figures ranges from 80% to 88% are reported for all languages.

1 Introduction

Stemming is critical for many NLP, IR and IE problems (Hull, 1996). In the current paper, we report construction of a semi supervised stemmer that does stemming by *minimizing the total number of distinct stems*. The input to the system is the word list along with the legal suffix list of the language. Even if a language does not have an elaborate linguistic tradition and exhaustive body of linguistic work, the language is expected to have at least the legal suffix list for nouns and verbs.

To get the intuition behind our work, consider the word list {*boy, boys, moss, mosses*}. The splitting (the *split* is formally defined later) that generate the minimum number of stems (*viz.*, 2) from the above word list is {*boy*+ϕ, *boy*+*s*, *moss*+ϕ, *moss*+*es* }, where ϕ is the null suffix. The minimum stem set is {*boy, moss*}. Any other splitting, say *mosse*+*s* will increase the number of stems.

This work is applicable to languages with concatenative morphology where suffixes stack one after another. However, problems arise when there are phonemic changes in the boundaries of the stems and suffixes (*sandhi*). In such situation, existence of fused, composite suffixes in the suffix list is assumed.

The roadmap of the paper is as follows. Related work in morphology learning is explained in section 2. Notations and terminologies used in this paper are defined in section 3. In section 4 we defined the stemming problem addressed in this work. Two models for this stemming are proposed in section 5 and section 6. In section 7, we described various experiments conducted. The conclusions and future works are presented in section 8.

2 Related Work

Morphology learning is one of the widely attempted problems in NLP. A recent survey by Harald Hammarström (2011) gives an overall view of unsupervised morphology learning. The Linguistica (Goldsmith, 2001) model based on minimum description length (MDL) principle is one of the benchmark works of unsupervised stemming. In the Linguistica model, the authors defined a signature structure. The objective of their stemming approach is the minimization of total description length, *i.e.*, the description length of stem list, suffix list, signatures and corpus.

Maximum a posteriori model (Creutz and Lagus, 2007) is a generalization of the Linguistica model, in the sense of being a recursive MDL. This probabilistic approach is more suitable for languages with more than one suffix. Stochastic transducer based model (Clark, 2001) and generative probabilistic model (Snover et al., 2002) are other relevant probabilistic models for stemming.

International Joint Conference on Natural Language Processing, pages 774–780,
Nagoya, Japan, 14-18 October 2013.

A Markov Random Field by Dreyer (2009) is also a useful probabilistic approach related to unsupervised morphology.

Graph based model (Johnson and Martin, 2003), lazy learning based model (van den Bosch and Daelemans, 1999), clustering based same stem identification model (Hammarström, 2006a; Hammarström, 2006b), ParaMor system for paradigm learning (Monson et al., 2008) and full morpheme segmentation and automatic induction of orthographic rules (Dasgupta and Ng, 2007; Dasgupta and Ng, 2006) are also a relevant.

3 Terminology and Notation

Let us define some terms and notations used throughout this paper.

w: word

W: input word list

N: number of words in the input word list

X: input suffix list

x: suffix candidate of w (possible suffix)

ϕ: null suffix, *i.e.*, the suffix with zero length.

t: stem candidate of w (possible stem)

T: the set of possible stems from the word list

$|\cdot|$: overloaded for the cardinality of a set and the length of a string

'$+$': splitting (breakage) of string

'$\cdot$': concatenation of strings

Stem: the longest common prefix of the inflected words of a lexeme. The stem from the set of inflections of the lexeme *play* ({*plays*, *playing* and *played*}) is *play*. Note that while a lexeme has to be a meaningful word in the language, the stem need not to be so. *E.g.*, stem of the words *lady* and *ladies* is *lad*, but the lexeme is *lady*.

Suffix: the portion(s) of the word after removing its stem. *E.g.*, the suffix of *boys* is *s*. The suffix can be a null string (ϕ) or chain of suffixes (Words in agglutinative languages can have multiple suffixes. In this case, chain of suffixes is taken as a single suffix.).

Split: the outcome of the process of segmentation (null strings permitted). A word can be segmented in multiple ways, giving rise to multiple *splits*. *E.g.*, a *split* of *boys* can be *b+oys*, *bo+ys*, *boy+s* or *boys+*ϕ. The *correct split* is the *split* that separates a word in to its correct stem and correct suffix. *E.g.*, the *boy+s* is the *correct split* of *boys*.

Splitset: a set of *splits* obtained from the whole input word list. For every word in the word list, exactly one *split* will be there in the *splitset*. In other words, $splitset = \{\ t + x \mid t \cdot x \in W$ and for any $t' + x' \in splitset$, $t \cdot x = t' \cdot x' \rightarrow t = t'$ and $x = x'\}$. *E.g.*, {*bo+ys*, *girl+*ϕ, *play+ing*} is a *splitset* of {*boys*, *girl*, *playing*}. The *correct splitset* is defined as the set of *correct splits* of all the words from the given word list. The *correct splitset* of {*boys*, *girl*, *playing*} is {*boy+s*, *girl+*ϕ, *play+ing*}.

$T_s(splitset)$: the set of stems from the $splitset$. *E.g.*, T_s({*bo+ys*, *girl+*ϕ, *play+ing*}) = {*bo*, *girl*, *play*}.

$X_s(splitset)$: the set of suffixes from the $splitset$. *E.g.*, X_s({*bo+ys*, *girl+*ϕ, *play+ing*}) = {*ys*, ϕ, *ing*}.

4 Problem Definition

The stemming problem addressed in this paper is defined as follows. Given a list of word forms W and a list of suffixes X, the problem is to find the $correct\ splitset$.

The suffix list of a language plays a crucial role in this problem. The suffix list considered in this problem should contain all atomic suffixes (*E.g.*, *s*, *es*, *ing*) and its orthographic variants (*E.g.*, *iness* in *happiness*). The suffix list also should contain chain of suffixes in case of agglutination. *E.g.*, the concatenated form of Malayalam[1] plural marker കൾ(*kal*) and genitive case marker ുടെ(*ude*) is കളുടെ(*kalude*). This concatenated form should be in the suffix list since it is the suffix (as per our definition) of the word കുട്ടികളുടെ(*kuttikalude*)(of children).

The suffix list X should be as large as possible and X should be a superset of all suffixes in the word list, *i.e.*, $X_s(correct\ splitset) \subseteq X$. *E.g.*, for the sample word list W = {*boy*, *boys*, *moss*, *mosses*}, the set $X_s(correct\ splitset)$ is { ϕ, *s*, *es*}, where ϕ is the null suffix. So the suffix list should contain at least ϕ, *s* and *es*.

The desired output of the above input is its *correct splitset*, *i.e.*, {*boy+*ϕ, *boy+s*, *moss+*ϕ,

[1] A morphologically rich language of India belonging to the Dravidian family.

moss+es}. Two computational models for this stemming problem is proposed in the section 5 and section 6.

5 Minimum Stem Set Model for Stemming

Consider the sample word list {*boy*, *boys*, *moss*, *mosses*} and suffix list {ϕ, *s*, *es*}. Out of all possible *splitsets* of this input, the *correct splitset*, {*boy*+ϕ, *boy*+*s*, *moss*+ϕ, *moss*+*es*} produces the minimum number of distinct stems. This intuition leads to the Minimum Stem Set model for stemming. The Minimum Stem Set model (MSS) identifies the *correct splitset* by minimizing the number of distinct stems. In other words, this model identifies the *splitset* with the minimum number of distinct stems as the *correct split*.

Core of the MSS model is an optimization problem (MSS problem). The MSS problem is formally stated as follows,

Input: A list of word forms (W) and a set of suffixes (X) such that $X_s(correct\ splitset) \subseteq X$

Output: $\underset{splitset:X_s(splitset)\subseteq X}{argmin} \left\{ |T_s(splitset)| \right\}$

5.1 Greedy Algorithm for MSS

Since the complexity of computing the MSS problem is NP-Hard (Vasudevan and Bhattacharyya, 2012), we designed an approximation algorithm by utilizing similarity between our problem and the set cover problem. Set cover problem is a well known NP-Hard problem, which has a simple greedy approximation algorithm with approximation factor of $log(N)$ (Chvatal, 1979). This approximation factor is the best attainable factor for the set cover problem (Feige, 1998). The corresponding greedy algorithm for MSS problem is the best polynomial time approximation algorithm. This greedy approximation algorithm for the MSS problem (Approx-MSS) is described below.

Input of the Approx-MSS algorithm is a word list W and a suffix list X. The algorithm first initializes a set of all possible stems T. This can be done by stripping suffixes in X from end of each word in W. Then it initialize sets of all possible inflections of each t in T, let's call $Infl(t)$. $Infl(t)$ can be initialized by appending suffixes from X to t. If a word created by appending a suffix x in X to t is in W, then add that word into $Infl(t)$. Set of all possible stems (T) and $Infl(t)$ of the running example is shown in Table 1.

After the initialization, the algorithm start the iterations with an empty $splitset$. In the first step, it chooses a stem t from T that has maximum $|Infl(t) \cap W|$. *I.e.*, it finds a $t^* = \underset{t\in T}{argmax}\{|Infl(t) \cap W|\}$. In the next step, for all words (w) in $Infl(t^*) \cap W$, the $split\ t^* + x$ is added to $splitset$, where x is the suffix of word w after the stem t^*. Then it removes all words from W whose $splits$ are added to $splitset$. This process is repeated until the *splitset* is complete, *i.e.,* for all words there is a $split$ in the $splitset$. The complexity of this approximation algorithm is $O(|W||X|)$ and approximation factor is $log(|W|)$.

Consider the example shown in Table 1. Initially both *boy* and *moss* have highest $|Infl(t) \cap W|$. So the greedy algorithm chooses either one of them in the first step as t^*. In the next step it chooses the other one. By these two steps, the greedy algorithm identifies the *correct split* of all four words.

T	*mosses*	*mosse*	*moss*	*mos*	*boys*	*boy*
$Infl(t)$	{*mosses*}	{*mosses*}	{*mosses*, *moss*}	{*moss*}	{*boys*}	{*boys*, *boy*}

Table 1: *Possible Stems, their* $Infl()$

6 Weighted Minimum Stem Set (WMSS) Model

MSS problem uses the information from other words to identify the stem of each word. If the word list doesn't have any other inflections of a word, then MSS cannot choose the stem properly. In this case MSS randomly selects one of the possible stems. This is one of the main drawbacks of MSS. Languages with poor morphology have a lesser number of inflections than that of language with rich morphology. So in a word list with fixed number of words, the above problem is more serious for morphologically poor languages.

We extended the MSS model to a Weighted Minimum Stem Set (WMSS) model, which reduces the number of distinct stems and the number of splits with non empty suffixes. Output of this model is also a *splitset*. Consider a small word list {*boy*, *boys*, *moss*, *mosses*} and a suffix list {ϕ, *s*, *es*, *ses*}. In this case both {*boy*+ϕ, *boy*+*s*, *moss*+ϕ,

moss+es} and {*boy+*ϕ, *boy+s*, *mos+s*, *mos+ses*} are optimum solutions for MSS problem. In such a tie situation, the WMSS model prefer the *splitset* with more number of null suffix (ϕ), *i.e.*, the first one. From our knowledge about English language, we can see that the first one is the *correct splitset*.

In the WMSS model, a weight function $wg(t)$ is defined for each and every possible stem t as $wg(t) = 1 + \frac{[t \notin W]}{|W|}$. Where $[t \notin W]$ is the Iverson bracket (Weisstein, Online 30 04 2010), *i.e.*, it is 1 if $t \notin W$, 0 otherwise. WMSS will find out a *splitset* such that the total weight of all stems in $T_s(splitset)$ is minimum. Let's define the problem in WMSS model formally.

Input: A list of word forms (W) and a set of suffixes (X) such that $X_s(correct\ splitset) \subseteq X$

Output: $$\underset{splitset:X_s(splitset)\subseteq X}{argmin} \left\{ \sum_{t \in T_s(splitset)} wg(t) \right\}$$

In this extended problem formulation, $wg(t)$ contain two terms. The first term, the constant 1 is for reducing the number of distinct stems and the second term, $\frac{[t \notin W]}{|W|}$ is for reducing the number of *splits* with non empty suffixes. If there is no second term then $wg(t)$ = 1 and it is exactly the same as MSS problem.

Since the maximum value of the second term in WMSS is $\frac{1}{|W|}$ and maximum number of stems in any $T_s(splitset)$ is less than $|W|$, $|T_s(splitset)| < \sum_{t \in T_s(splitset)} wg(t) < |T_s(splitset)| + 1$. Therefore any solution of WMSS should be a solution of MSS, but the reverse is false. Relevance of this WMSS problem comes only if there are multiple solutions for MSS problem.

Since the solution of WMSS problem is a solution of MSS problem, the reduction from MSS to WMSS is trivial. Suppose WMSS have a polynomial time algorithm, then we can use that algorithm for MSS problem also. Since MSS is NP-Hard we can say that, WMSS is also NP-Hard.

6.1 Greedy Algorithm for WMSS

The WMSS problem can be solved effectively by utilizing its similarity with weighted set cover problem. Weighted set cover problem is also an NP-Hard problem, and its greedy approximation have a bound of $log(N)$. The greedy algorithm for weighted set cover problem is adapted for WMSS problem. The corresponding greedy algorithm for WMSS (Approx-WMSS) is explained below.

The Approx-WMSS is similar to Approx-MSS. The only difference is in the first step. While Approx-MSS algorithm selects a stem with maximum $|Infl(t) \cap W|$ in the first step, Approx-WMSS algorithm selects a stem with maximum $\frac{|Infl(t) \cap W|}{wg(t)}$. Note that, when $wg(t)$ is 1 then both terms are the same. All remaining steps are the same for both algorithms. Similar to Approx-MSS algorithm, the complexity of this approximation algorithm is $O(|W||X|)$ and approximation factor is $log(|W|)$.

Consider the W = {*boy*, *boys*, *moss*, *mosses*} and X = {ϕ, *s*, *es*, *ses*}. The set of all possible stems and its corresponding $Infl()$ and $wg()$ are shown in Table 2. Initially *boy* has the highest $\frac{|Infl(t) \cap W|}{wg(t)}$. So this greedy algorithm chooses the stem *boy* and add *boy+*ϕ and *boy+s* to *splitset* in the first step. In the next step it chooses *moss* and add *moss+es* and *moss+*ϕ to *splitset*. By these two steps, this greedy algorithm terminate by identifying *correct splitset*.

T	*mosses*	*mosse*	*moss*	*mos*	*boys*	*boy*
$Infl(t)$	{*mosses*}	{*mosses*}	{*mosses*, *moss*}	{*moss*}	{*boys*}	{*boys*, *boy*}
$wg(t)$	1	$1+\frac{1}{4}$	1	$1+\frac{1}{4}$	1	1

Table 2: *Possible Stems, their* $Infl()$ *and* $wg()$

7 Experimentation

Two new stemming systems based on the greedy algorithms for MSS problem and WMSS problem are implemented. Performances of these systems are evaluated for four languages from Indo-European family and Dravidian family. The selected languages are English, Hindi, Marathi and Malayalam, in the increasing order of morphological complexity. First three languages are from Indo-European family while the fourth language, Malayalam is a highly agglutinative language from Dravidian family. These spectrum of languages from different families with different morphological richness is necessary for the evaluation of the suitability of proposed models.

Performance of proposed models are compared with different baselines. The first baseline is a random stem selection, which randomly selects a *split* for each word such that the suffix in this *split* is in the input suffix list. The length of the suffix (or

stem) is another information that can provide second and third baselines. The second one selects a *split* for each word form that has the smallest stem, albeit with the suffix in the input suffix list. Similarly the third one selects the *split* with the largest stem.

Linguistica is an MDL based system that identifies stem of each word in a word list without using any other input. One of the heuristics used in Linguistica model is modified to make the fourth baseline. In the Linguistica heuristics, a probability is assigned to every *split* for every word. Then iteratively it learns the best probability distribution by optimizing a figure of merit, which is a function of length and frequency of morphemes. Since there is no need to consider any *split* with a suffix which is not in the input suffix list, the sample space can be minimized. Probability distribution after this modification is learned using the same iterative procedure as in Linguistica. We implemented this modified Linguistica algorithm and considered it as fourth baseline.

7.1 Data Analysis

Word list of size 10,000 distinct words in Unicode format were selected for English, Hindi, Marathi and Malayalam. English words are taken from Brown and BNC corpora (Francis and Kucera, 1964; Edition, 2007). Selected Hindi words are from tourism and news corpus. The source of Marathi words for experimentation is the corpora from the Indian Language Corpora Initiative (ILCI) project, which is a Government of India effort (http://www.tdil.mit.gov.in). Malayalam words are obtained from IIITMK[2] and from various blogs and newspapers. For each words the correct stem as per the definition, *i.e.*, the largest prefix of all inflected forms of the lexeme, is identified for the evaluation. Suffix lists are mainly created from the words in the word list. By adding available suffixes from web, the suffix lists are expanded as big as possible.

Counts and frequencies of stems and suffixes are relevant statistics to reflect the nature of word list for stemming. So the number of distinct stems (*StCount*) and suffixes (*SfCount*) are counted from each word list. The average stem frequencies (*StFreq*) and average suffix frequencies (*SfFreq*) are also measured from word lists of all four languages. These measured values are shown in Table 3.

Language	*StCount*	*StFreq*	*SfCount*	*SfFreq*	$\|X\|$
English	4974	2.01	43	232.58	436
Hindi	4792	2.09	134	74.63	726
Marathi	4086	2.45	604	16.56	1958
Malayalam	1077	9.29	762	13.12	26248

Table 3: Statistics of Word List and Suffix List (X)

Number of distinct stems in the word form list decreases and average stem frequencies increases along with morphological complexity of language. Similarly the number of distinct suffixes in the word list increases and average suffix frequencies decreases along with morphological complexity. We can also see that the number of suffixes in a language also increases with morphological complexity. Since these patterns are quite intuitive, the data taken for experiments seems to be proper samples that represents the languages.

7.2 Results and Discussion

The accuracy of four baselines and two newly proposed systems for four languages are tabulated in Table 4. The results indicates the effectiveness of the new systems over baseline systems across various languages. Improvement in the performance of WMSS over MSS is also clearly visible in the table. Above 80% accuracies for all languages are obtained by using the WMSS model. English, Hindi, Marathi and Malayalam are the languages in the increasing order of morphological complexity. We can observe that the accuracies are decreasing along with the morphological complexity of language. This indicates stemming is difficult for morphologically complex languages.

Language	Random Stem	Largest Stem	Smallest Stem	Modified Linguistica	MSS	WMSS
English	44.98	47.39	49.36	53.82	84.44	88.86
Hindi	50.68	43.04	57.44	62.74	80.71	83.98
Marathi	41.66	30.44	69.766	59.33	78.28	80.19
Malayalam	19.31	3.51	57.58	65.86	78.32	80.06

Table 4: Stemming Accuracies in Percentage

For all four languages, the baseline which selects stems with maximum length have a lesser score than the baseline which selects stems with minimum length. This shows, if there are more

[2]Indian Institute of Information Technology and Management-Kerala

than one stem candidate, then smaller stems is preferred. Since null suffix is present in the suffix list, maximum stem length baseline always selects the word itself as its stem. So the low score for maximum stem length baseline for Malayalam indicates, most of the Malayalam words are in the inflected form. To get better insight about the remaining issues an error analysis of the output sample is required.

7.3 Error Analysis

To get better insight about the remaining issues, erroneous samples generated by the best performing system, *i.e.*, WMSS, are categorized in to under stemming [3], over stemming [4] and weight error. The weight error is the case where the correct stem is in the word list but the identified incorrect stem is not. Such errors can be corrected by modifying the weight function in the WMSS formulation. Percentage of errors in various categories are tabulated in Table 5.

If the number of suffixes in the word list is very small compared to the total number of suffixes in the suffix list, then there is a high chance for over-stemming. So the ratio between total number of suffixes and the number of suffixes in the word list (suffix ratio), for all four languages are also included in Table 5.

Language	Under-Stemming	Over-Stemming	Weight-Error	Suffix ratio
English	2.76	8.38	3.74	10.0
Hindi	3.90	12.12	5.94	5.42
Marathi	8.36	11.45	5.57	3.24
Malayalam	0.93	19.01	0.24	34.5

Table 5: Percentage of Errors and Suffix ratios

Suffix ratio of English is high (10). It decreases in Hindi, and further decreases in Marathi. According to this pattern, the under stemming errors are very few (only 3%) in English and it increases in Hindi and Marathi. The suffix ratio of Malayalam is higher than English so the under stemming errors are negligibly small (less than 1%). The relation between suffix ratio and under stemming errors are clearly visible from these numbers. So to reduce the under stemming errors, we need to increase the number of input suffixes.

[3] identified stem is longer than correct stem, *e.g.*, *mosse* in *mosses*

[4] identified stem is shorter than correct stem, *e.g.*, *s* in *sing*

The over stemming errors increases from English to Malayalam. This indicates that the over stemming errors are more sensitive to morphological complexity than the suffix ratio. From the table we can see that, weight errors are significant except in Malayalam. This indicates the requirement of weight modification. Also, we can see that the weight errors are high in Hindi and Marathi, and hence the weight modification is crucial for these languages.

After the analysis, the main observation is about the importance of weight modification. Some sample words from all four languages are shown in Figure 1.

Language	Word	Identified Stem	Correct Stem (in case of erroneous sample)
English	pretenses halfways rhodes.	pretenses halfway rhodes	pretense
Hindi	मच्छर (machhar) (mosquito) चूर्णो (choornom) सोफे (sophe)	मच्छ (machha) चूर्ण (choorn) सोफ (sofa)	मच्छर (machhar)
Marathi	छेडतील (chhedtheel) (to provoke) सांभाळतात (saambhaalthaath) (to look after) दिव्य (divya) (magnificient)	छेड (chhed) सांभाळ (sambhaal) दिव (diva)	दिव्य (divya)
Malayalam	ക്കൂണിക്കുമെല്ലാം (koonilumellam) (also in mushroom) പാമ്പിനുള്ളിൽ (paampinullil) (inside snake) അണുക്കളും (anukkalum) (and atoms)	ക്കൂണ (koona) പാമ്പ (paampa) അണ (ana)	അണു(anu)

Figure 1: Output Samples (WMSS)

8 Conclusion

Two algorithms for stemming, that produces a mapping from words to stems by minimizing the number of stems upto a limit, given a word list and a suffix list are proposed and implemented. Stemming systems that use these algorithms are evaluated using languages from Indo European and Dravidian families. Moderate to high accuracies of stemming are obtained in case of for all four languages: English, Hindi, Malayalam and Marathi.

Collecting a word list is relatively an easy task for a new language. But, collecting a complete list of suffixes is a much more involved task since detailed linguistic work is required. So completely unsupervised stemming is our future work. Stems will be produced from only the word form list.

References

V. Chvatal. 1979. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):pp. 233–235.

Alexander Clark. 2001. Partially supervised learning of morphology with stochastic transducers. In *NLPRS*, pages 341–348.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *TSLP*, 4(1).

Sajib Dasgupta and Vincent Ng. 2006. Unsupervised morphological parsing of bengali. *Language Resources and Evaluation*, pages 311–330.

Sajib Dasgupta and Vincent Ng. 2007. High-performance, language-independent morphological segmentation. In *HLT-NAACL*, pages 155–163.

Markus Dreyer and Jason Eisner. 2009. Graphical models over multiple strings. In *Proc. of EMNLP-09*, pages 101–110.

The British National Corpus, Version 3 BNC XML Edition. 2007. Distributed by Oxford University Computing Services on behalf of the BNC Consortium.

Uriel Feige. 1998. A threshold of ln n for approximating set cover. *JOURNAL OF THE ACM*, 45:314–318.

Withrop N. Francis and Henry Kucera. 1964. *Manual of Information to accompany A standard corpus of present-day edited American English, for use with digital computers with Digital Computers.* Brown University Press.

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

Harald Hammarström and Lars Borin. 2011. Unsupervised learning of morphology. *CL*, (2):309–350.

Harald Hammarström. 2006a. A naive theory of affixation and an algorithm for extraction. In *Proc. of HLT-NAACL-06*, pages 79–88, June.

Harald Hammarström. 2006b. Poor man's stemming: Unsupervised recognition of same-stem words. In *AIRS*, pages 323–337.

David A. Hull. 1996. Stemming algorithms: A case study for detailed evaluation. *JASIS*, 47(1):70–84.

Howard Johnson and Joel Martin. 2003. Unsupervised learning of morphology for english and inuktitut. In *Proc. of NAACL-HLT-03*, pages 43–45.

Christian Monson, Jaime G. Carbonell, Alon Lavie, and Lori S. Levin. 2008. Paramor and morpho challenge 2008. In *CLEF*, pages 967–974.

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 *Proc. of ACL-WMPL-02*, pages 11–20.

Antal van den Bosch and Walter Daelemans. 1999. Memory-based morphological analysis. In *Proc. of ACL-99*.

N. Vasudevan and Pushpak Bhattacharyya. 2012. Optimal stem identification in presence of suffix list. In *CICLing (1)*, pages 92–103.

Eric W Weisstein. [Online; 30-04-2010]. Iverson bracket. MathWorldA Wolfram Web Resources. http://mathworld.wolfram.com/IversonBracket.html.