In: *Proceedings of CoNLL-2000 and LLL-2000*, pages 67–72, Lisbon, Portugal, 2000.

Knowledge-Free Induction of Morphology Using Latent Semantic Analysis

Patrick Schone and **Daniel Jurafsky**
University of Colorado
Boulder, Colorado 80309
{schone, jurafsky}@cs.colorado.edu

Abstract

Morphology induction is a subproblem of important tasks like automatic learning of machine-readable dictionaries and grammar induction. Previous morphology induction approaches have relied solely on statistics of hypothesized stems and affixes to choose which affixes to consider legitimate. Relying on stem-and-affix statistics rather than semantic knowledge leads to a number of problems, such as the inappropriate use of valid affixes ("ally" stemming to "all"). We introduce a semantic-based algorithm for learning morphology which only proposes affixes when the stem and stem-plus-affix are sufficiently similar semantically. We implement our approach using Latent Semantic Analysis and show that our semantics-only approach provides morphology induction results that rival a current state-of-the-art system.

1 Introduction

Computational morphological analyzers have existed in various languages for years and it has been said that "the quest for an efficient method for the analysis and generation of word-forms is no longer an academic research topic" (Karlsson and Karttunen, 1997). However, development of these analyzers typically begins with human intervention requiring time spans from days to weeks. If it were possible to build such analyzers automatically without human knowledge, significant development time could be saved.

On a larger scale, consider the task of inducing machine-readable dictionaries (MRDs) using *no* human-provided information ("knowledge-free"). In building an MRD, "simply expanding the dictionary to encompass every word one is ever likely to encounter...fails to take advantage of regularities" (Sproat, 1992, p. xiii). Hence, automatic morphological analysis is also critical for selecting appropriate and non-redundant MRD headwords.

For the reasons expressed above, we are interested in knowledge-free morphology induction. Thus, in this paper, we show how to automatically induce morphological relationships between words.

Previous morphology induction approaches (Goldsmith, 1997, 2000; DéJean, 1998; Gaussier, 1999) have focused on inflectional languages and have used statistics of hypothesized stems and affixes to choose which affixes to consider legitimate. Several problems can arise using only stem-and-affix statistics: (1) valid affixes may be applied inappropriately ("ally" stemming to "all"), (2) morphological ambiguity may arise ("rating" conflating with "rat" instead of "rate"), and (3) non-productive affixes may get accidentally pruned (the relationship between "dirty" and "dirt" may be lost).[1]

Some of these problems could be resolved if one could incorporate word semantics. For instance, "all" is not semantically similar to "ally," so with knowledge of semantics, an algorithm could avoid conflating these two words. To maintain the "knowledge-free" paradigm, such semantics would need to be automatically induced. Latent Semantic Analysis (LSA) (Deerwester, *et al.*, 1990); Landauer, *et al.*, 1998) is a technique which automatically identifies semantic information from a corpus. We here show that incorporating LSA-based semantics *alone* into the morphology-induction process can provide results that rival a state-of-the-art system based on stem-and-affix statistics (Goldsmith's *Linguistica*).

[1] Error examples are from Goldsmith's *Linguistica*

Our algorithm automatically extracts potential affixes from an untagged corpus, identifies word pairs sharing the same proposed stem but having different affixes, and uses LSA to judge semantic relatedness between word pairs. This process serves to identify valid morphological relations. Though our algorithm could be applied to any inflectional language, we here restrict it to English in order to perform evaluations against the human-labeled CELEX database (Baayen, *et al.*, 1993).

2 Previous work

Existing induction algorithms all focus on identifying prefixes, suffixes, and word stems in inflectional languages (avoiding infixes and other language types like concatenative or agglutinative languages (Sproat, 1992)). They also observe high frequency occurrences of some word endings or beginnings, perform statistics thereon, and propose that some of these appendages are valid morphemes.

However, these algorithms differ in specifics. DéJean (1998) uses an approach derived from Harris (1951) where word-splitting occurs if the number of distinct letters that follows a given sequence of characters surpasses a threshold. He uses these hypothesized affixes to resegment words and thereby identify additional affixes that were initially overlooked. His overall goal is different from ours: he primarily seeks an affix inventory.

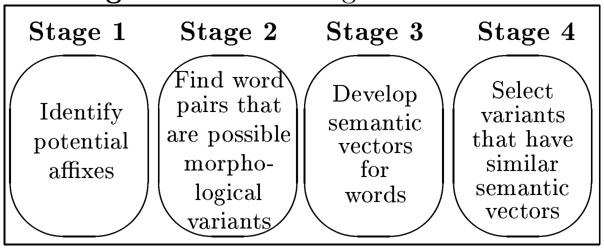
Goldsmith (1997) tries cutting each word in exactly one place based on probability and lengths of hypothesized stems and affixes. He applies the EM algorithm to eliminate inappropriate parses. He collects the possible suffixes for each stem calling these a *signature* which aid in determining word classes. Goldsmith (2000) later incorporates minimum description length to identify stemming characteristics that most compress the data, but his algorithm otherwise remains similar in nature. Goldsmith's algorithm is practically knowledge-free, though he incorporates capitalization removal and some word segmentation.

Gaussier (1999) begins with an inflectional lexicon and seeks to find derivational morphology. The words and parts of speech from his inflectional lexicon serve for building relational families of words and identifying sets of word pairs and suffixes therefrom. Gaussier splits words based on *p-similarity* – words that agree in exactly the first p characters. He also builds a probabilistic model which indicates that the probability of two words being morphological variants is based upon the probability of their respective changes in orthography and morphosyntactics.

3 Current approach

Our algorithm also focuses on inflectional languages. However, with the exception of word segmentation, we provide it no human information and we consider only the impact of semantics. Our approach (see Figure 1) can be decomposed into four components: (1) initially selecting candidate affixes, (2) identifying affixes which are potential morphological variants of each other, (3) computing semantic vectors for words possessing these candidate affixes, and (4) selecting as valid morphological variants those words with similar semantic vectors.

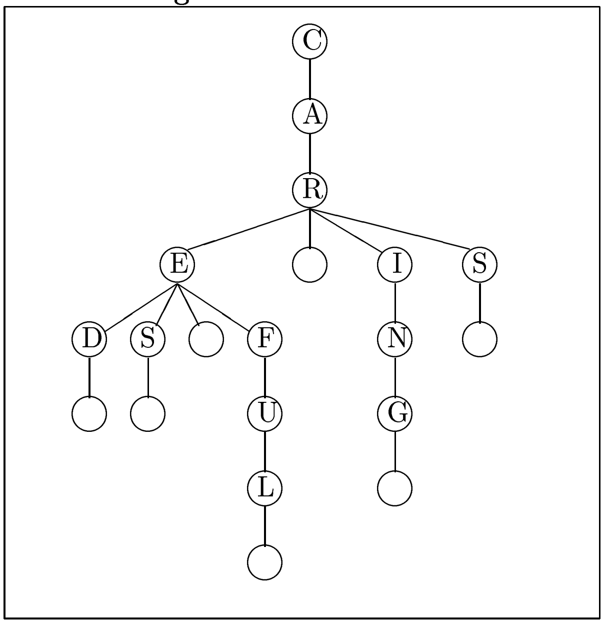
Figure 1: Processing Architecture



3.1 Hypothesizing affixes

To select candidate affixes, we, like Gaussier, identify p-similar words. We insert words into a *trie* (Figure 2) and extract potential affixes by observing those places in the trie where branching occurs. Figure 2's hypothesized suffixes are NULL, "s," "ed," "es," "ing," "e," and "eful." We retain only the K most-frequent candidate affixes for subsequent processing. The value for K needs to be large enough to account for the number of expected regular affixes in any given language as well as some of the more frequent irregular affixes. We arbitrarily chose K to be 200 in our system. (It should also be mentioned that we can identify potential *prefixes* by inserting words into the trie in reversed order. This prefix mode can additionally serve for identifying capitalization.)

Figure 2: Trie structure



3.2 Morphological variants

We next identify pairs of candidate affixes that descend from a common ancestor node in the trie. For example, ("s", NULL) constitutes such a pair from Figure 2. We call these pairs *rules*.

Two words sharing the same root and the same affix rule, such as "cars" and "car," form what we call a *pair of potential morphological variants* (PPMVs). We define the *ruleset* of a given rule to be the set of all PPMVs that have that rule in common. For instance, from Figure 2, the ruleset for ("s", NULL) would be the pairs "cars/car" and "cares/care." Our algorithm establishes a list which identifies the rulesets for every hypothesized rule extracted from the data and then it must proceed to determine which rulesets or PPMVs describe true morphological relationships.

3.3 Computing Semantic Vectors

Deerwester, *et al.* (1990) showed that it is possible to find significant semantic relationships between words and documents in a corpus with virtually no human intervention (with the possible exception of a human-built stop word list). This is typically done by applying singular value decomposition (SVD) to a matrix, M, where each entry $M(i,j)$ contains the frequency of word i as seen in document j of the corpus. This methodology is referred to as Latent Semantic Analysis (LSA) and is well-described in the literature (Landauer, *et al.*, 1998; Manning and Schütze, 1999).

SVDs seek to decompose a matrix A into the product of three matrices U, D, and V^T where U and V^T are orthogonal matrices and D is a diagonal matrix containing the singular values (squared eigenvalues) of A. Since SVD's can be performed which identify singular values by descending order of size (Berry, *et al.*, 1993), LSA truncates after finding the k largest singular values. This corresponds to projecting the vector representation of each word into a k-dimensional subspace whose axes form k (latent) semantic directions. These projections are precisely the rows of the matrix product $U_k D_k$. A typical k is 300, which is the value we used.

However, we have altered the algorithm somewhat to fit our needs. First, to stay as close to the knowledge-free scenario as possible, we neither apply a stopword list nor remove capitalization. Secondly, since SVDs are more designed to work on normally-distributed data (Manning and Schütze, 1999, p. 565), we operate on Z-scores rather than counts. Lastly, instead of generating a term-document matrix, we build a term-term matrix.

Schütze (1993) achieved excellent performance at classifying words into quasi-part-of-speech classes by building and performing an SVD on an Nx$4N$ term-term matrix, $M(i,Np+j)$. The indices i and j represent the top N highest frequency words. The p values range from 0 to 3 representing whether the word indexed by j is positionally offset from the word indexed by i by -2, -1, +1, or +2, respectively. For example, if "the" and "people" were respectively the 1st and 100th highest frequency words, then upon seeing the phrase "the people," Schütze's approach would increment the counts of $M(1,2N+100)$ and $M(100,N+1)$.

We used Schütze's general framework but tailored it to identify local *semantic* information. We built an Nx$2N$ matrix and our p values correspond to those words whose offsets from word i are in the intervals [-50,-1] and [1,50], respectively. We also reserve the Nth position as a catch-all position to account for all words that are not in the top (N-1). An important issue to resolve is how large should N be. We would like to be able to incorporate semantics for an arbitrarily large number of words and LSA quickly becomes impractical on large sets. Fortunately, it is possible to build a matrix with a smaller value of N (say, 2500), perform an SVD thereon, and then fold in remaining terms (Manning and Schütze, 1999, p. 563). Since the U and V matrices of an SVD are orthogonal matrices, then UU^T=VV^T=I. This implies that AV=UD.

This means that for a new word, w, one can build a vector ξ_w^T which identifies how w relates to the top N words according to the p different conditions described above. For example, if w were one of the top N words, then ξ_w^T would simply represent w's particular row from the A matrix. The product $\mathbf{\Omega_w}$= $\xi_w^T V_k$ is the projection of ξ_w^T into the k-dimensional latent semantic space. By storing an index to the words of the corpus as well as a sorted list of these words, one can efficiently build a set of semantic vectors which includes each word of interest.

3.4 Statistical Computations

Morphologically-related words frequently share similar semantics, so we want to see how well semantic vectors of PPMVs correlate. If we know how PPMVs correlate in comparison to other word pairs from their same rulesets, we can actually determine the semantic-based probability that the variants are legitimate. In this section, we identify a measure for correlating PPMVs and illustrate how ruleset-based statistics help identify legitimate PPMVs.

3.4.1 Semantic Correlation of Words

The cosine of the angle between two vectors $\mathbf{v_1}$ and $\mathbf{v_2}$ is given by,

$$cos(\mathbf{v_1},\mathbf{v_2}) = \frac{\mathbf{v_1}\cdot\mathbf{v_2}}{\parallel \mathbf{v_1} \parallel \parallel \mathbf{v_2} \parallel}.$$

We want to determine the correlation between each of the words of every PPMV. We use what we call a *normalized cosine score* (NCS) as a correlation. To obtain a NCS, we first calculate the cosine between each semantic vector, $\mathbf{\Omega_w}$, and the semantic vectors from 200 randomly chosen words. By this means we obtain w's correlation mean (μ_w) and standard deviation (σ_w). If v is one of w's variants, then we define the NCS between $\mathbf{\Omega_w}$ and $\mathbf{\Omega_v}$ to be

$$\min_{y\epsilon\{w,v\}}\left(\frac{\cos(\mathbf{\Omega_w},\mathbf{\Omega_v})-\mu_y}{\sigma_y}\right).$$

Table 1 provides normalized cosine scores for several PPMVs from Figure 2 and from among words listed originally as errors in other systems. (NCSs are effectively Z-scores.)

Table 1: Normalized Cosines for various PPMVs

PPMVs	NCSs	PPMVs	NCSs
car/cars	5.6	ally/allies	6.5
car/caring	-0.71	ally/all	-1.3
car/cares	-0.14	dirty/dirt	2.4
car/cared	-0.96	rating/rate	0.97

3.4.2 Ruleset-level Statistics

By considering NCSs for all word pairs coupled under a particular rule, we can determine semantic-based probabilities that indicate which PPMVs are legitimate. We expect random NCSs to be normally-distributed according to $\mathcal{N}(0,1)$. Given that a particular ruleset contains n_R PPMVs, we can therefore approximate the number (n_T), mean (μ_T) and standard deviation (σ_T) of *true* correlations. If we define $\Phi_Z(\mu,\sigma)$ to be $\int_Z^\infty e^{-(\frac{X-\mu}{\sigma})^2}dx$, then we can compute the probability that the particular correlation is legitimate:

$$Pr(true) = \frac{n_T\,\Phi_Z(\mu_T,\sigma_T)}{(n_R-n_T)\,\Phi_Z(0,1)+n_T\,\Phi_Z(\mu_T,\sigma_T)}.$$

3.4.3 Subrules

It is possible that a rule can be hypothesized at the trie stage that is true under only certain conditions. A prime example of such a rule is ("es", NULL). Observe from Table 1 that the word "cares" poorly correlates with "car." Yet, it is true that "-es" is a valid suffix for the words "flashes," "catches," "kisses," and many other words where the "-es" is preceded by a voiceless sibilant.

Hence, there is merit to considering subrules that arise while performing analysis on a particular rule. For instance, while evaluating the ("es", NULL) rule, it is desirable to also consider potential subrules such as ("ches", "ch") and ("tes", "t"). One might expect that the average NCS for the ("ches", "ch") subrule might be higher than the overall rule ("es", NULL) whereas the opposite will likely be true for ("tes", "t"). Table 2 confirms this.

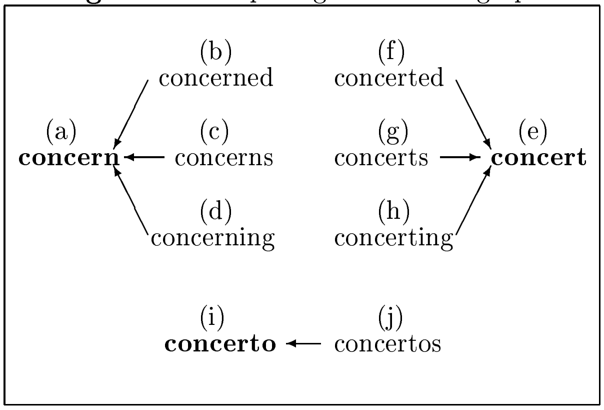
Table 2: Analysis of subrules

Rule/Subrule	Average	StDev	#instances
("es", NULL)	1.62	2.43	173
("ches","ch")	2.20	1.66	32
("shes", "sh")	2.39	1.52	15
("res", "r")	-0.69	0.47	6
("tes","t")	-0.58	0.93	11

4 Results

We compare our algorithm to Goldsmith's *Linguistica* (2000) by using CELEX's (Baayen, *et al.*, 1993) suffixes as a gold standard. CELEX is a hand-tagged, morphologically-analyzed database of English words. CELEX has limited coverage of the words from our data set (where our data consists of over eight million words from random subcollections of TREC data (Voorhees, et al,1997/8)), so we only considered words with frequencies of 10 or more.

Figure 3: Morphological directed graphs



Morphological relationships can be represented graphically as directed graphs (see Figure 3, where three separate graphs are depicted). Developing a scoring algorithm to compare directed graphs is likely to be prone to disagreements. Therefore, we score only the vertex sets of directed graphs. We will refer to these vertex sets as *conflation sets*. For example, *concern*'s conflation set contains itself as well as "concerned," "concerns," and "concerning" (or, in shorthand notation, the set is {a,b,c,d}).

To evaluate an algorithm, we sum the number of correct ($\mathcal{C}$), inserted ($\mathcal{I}$), and deleted ($\mathcal{D}$) words it predicts for each hypothesized conflation set. If X_w represents word w's conflation set according to the algorithm, and if Y_w represents its CELEX-based conflation set, then

$\mathcal{C} = \sum_{\forall w}(|X_w \bigcap Y_w|/|Y_w|)$,
$\mathcal{D} = \sum_{\forall w}(|Y_w - (X_w \bigcap Y_w)|/|Y_w|)$, and
$\mathcal{I} = \sum_{\forall w}(|X_w - (X_w \bigcap Y_w)|/|Y_w|)$.

However, in making these computations, we disregard any CELEX words that are not in the algorithm's data set and vice versa.

For example, suppose two algorithms were being compared on a data set where all the words from Figure 3 were available except "concerting" and "concertos." Suppose further that one algorithm proposed that {a,b,c,d,e,f,g,i} formed a single conflation set whereas the other algorithm proposed the three sets {a,b,c,d},{e,g,i}, and {f}. Then Table 3 illustrates how the two algorithms would be scored.

Table 3: Example of scoring

	a	b	c	d	e	f	g	i	Total
$\mathcal{C}1$	4/4	4/4	4/4	4/4	3/3	3/3	3/3	1/1	8
$\mathcal{D}1$	0/4	0/4	0/4	0/4	0/3	0/3	0/3	0/1	0
$\mathcal{I}1$	4/4	4/4	4/4	4/4	5/3	5/3	5/3	7/1	16
$\mathcal{C}2$	4/4	4/4	4/4	4/4	2/3	2/3	1/3	1/1	20/3
$\mathcal{D}2$	0/4	0/4	0/4	0/4	1/3	1/3	2/3	0/1	4/3
$\mathcal{I}2$	0/4	0/4	0/4	0/4	1/3	1/3	0/3	2/1	8/3

To explain Table 3, consider algorithm one's entries for 'a.' Algorithm one had proposed that X_a={a,b,c,d,e,f,g,i} when in reality, Y_a={a,b,c,d}. Since $|X_a \bigcap Y_a| = 4$ and $|Y_a|$=4, then $\mathcal{C}_A$=4/4. The remaining values of the table can be computed accordingly.

Using the values from Table 3, we can also compute precision, recall, and F-Score. Precision is defined to be $\mathcal{C}/(\mathcal{C}+\mathcal{I})$, recall is $\mathcal{C}/(\mathcal{C}+\mathcal{D})$, and F-Score is the product of precision and recall divided by the average of the two. For the first algorithm, the precision, recall, and F-Score would have respectively been 1/3, 1, and 1/2. In the second algorithm, these numbers would have been 5/7, 5/6, and 10/13.

Table 4 uses the above scoring mechanism to compare between *Linguistica* and our system (at various probability thresholds). Note that since *Linguistica* removes capitalization, it will have a different total word count than our system.

Table 4: Performance on English CELEX

Algorithm	*Linguistica*	LSA-based pr≥ 0.5	LSA-based pr≥ 0.7	LSA-based pr≥ 0.85
#Correct	10515	10529	10203	9863
#Inserts	2157	1852	1138	783
#Deletes	2571	2341	2667	3007
Precision	83.0%	85.0%	90.0%	92.6%
Recall	80.4%	81.8%	79.3%	76.6%
F-Score	81.6%	83.4%	**84.3%**	83.9%

5 Conclusions

These results suggest that semantics and LSA can play a key part in knowledge-free morphology induction. Semantics alone worked at least as well as Goldsmith's frequency-based approach. Yet we believe that semantics-based and frequency-based approaches play complementary roles. In current work, we are examining how to combine these two approaches.

References

Albright, A. and B. P. Hayes. 1999. An automated learner for phonology and morphology. Dept. of Linguistics, UCLA. At http://www.humnet.ucla.edu/humnet/linguistics/people/hayes/learning/learner.pdf.

Baayen, R. H., R. Piepenbrock, and H. van Rijn. 1993. The CELEX lexical database (CD-ROM), Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA.

Berry, M., T. Do, G. O'Brien, V. Krishna, and S. Varadhan. 1993. SVDPACKC user's guide. CS-93-194, University of Tennessee.

Déjean, H. 1998. Morphemes as necessary concepts for structures: Discovery from untagged corpora. University of Caen-Basse Normandie. http://www.info.unicaen.fr/~DeJean/travail/articles/pg11.htm.

Deerwester, S., S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science.*

Gaussier, É. 1999. Unsupervised learning of derivational morphology from inflectional lexicons. *ACL '99 Workshop Proceedings: Unsupervised Learning in Natural Language Processing*, University of Maryland.

Goldsmith, J. 1997. Unsupervised learning of the morphology of a natural language. University of Chicago.

Goldsmith, J. 2000. Unsupervised learning of the morphology of a natural language. University of Chicago. http://humanities.uchicago.edu/faculty/goldsmith.

Harris, Z. 1951. *Structural Linguistics.* University of Chicago Press.

Hull, D. A. and G. Grefenstette. 1996. A detailed analysis of English stemming algorithms. XEROX Technical Report, http://www.xrce.xerox.com/publis/mltt/mltt-023.ps.

Krovetz, R. 1993. Viewing morphology as an inference process. *Proceedings of the 16thACM/SIGIR Conference*, pp. 191-202.

Jurafsky, D. S. and J. H. Martin. 2000. *Speech and Language Processing.* Prentice Hall, Inc., Englewood, N.J.

Karlsson, F. and L. Karttunen,. 1997. "Subsentencial Processing." In *Survey of the State of the Art in Human Language Technology*, R. Cole, Ed., Giardini Editori e Stampatori, Italy.

Koskenniemi, K. 1983. *Two-level Morphology: a General Computational Model for Word-Form Recognition and Production.* Ph.D. thesis, University of Helsinki.

Landauer,T. K., P. W. Foltz, and D. Laham. 1998. Introduction to Latent Semantic Analysis. *Discourse Processes.* Vol. 25, pp. 259-284.

Lovins, J. 1968. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, Vol. 11, pp.22-31

Manning, C. D. and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*, MIT Press, Cambridge, MA.

Porter, M. 1980. An algorithm for suffix stripping. *Program*, Vol. 14(3), pp.130-137.

Ritchie, G. and G. J. Russell. 1992. *Computational morphology: Practical Mechanisms for the English Lexicon.* MIT.

Schütze, H. 1993. Distributed syntactic representations with an application to part-of-speech tagging. *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1504-1509.

Scott, D. 1992. *Multivariate Density Estimation: Theory, Practice, and Visualization.* John Wiley & Sons, New York.

Sproat, R. 1992. *Morphology and Computation.* MIT Press, Cambridge, MA.

Van den Bosch, A. and W. Daelemans. 1999. Memory-based morphological analysis. *Proc. of the 37th Annual Meeting of the ACL*, University of Maryland, pp. 285-292.

Voorhees, E., D. Hoffman, and C. Barnes. 1996-7. TREC Information Retrieval: Text Research Collection, Vols. 4-5 (CD-ROM), National Institute of Standards and Technology.

Woods, W. 2000. Aggressive morphology for robust lexical coverage. *Proceedings of the 6th ANLP/1st NAACL*, Seattle, WA.