Knowledge-Free Induction of Inflectional Morphologies

Patrick SCHONE
University of Colorado at Boulder
Boulder, Colorado 80309
schone@cs.colorado.edu

Daniel JURAFSKY
University of Colorado at Boulder
Boulder, Colorado 80309
jurafsky@cs.colorado.edu

Abstract

We propose an algorithm to automatically induce the morphology of inflectional languages using only text corpora and no human input. Our algorithm combines cues from orthography, semantics, and syntactic distributions to induce morphological relationships in German, Dutch, and English. Using CELEX as a gold standard for evaluation, we show our algorithm to be an improvement over any knowledge-free algorithm yet proposed.

1 Introduction

Many NLP tasks, such as building machine-readable dictionaries, are dependent on the results of morphological analysis. While morphological analyzers have existed since the early 1960s, current algorithms require human labor to build rules for morphological structure. In an attempt to avoid this labor-intensive process, recent work has focused on machine-learning approaches to induce morphological structure using large corpora.

In this paper, we propose a knowledge-free algorithm to automatically induce the morphology structures of a language. Our algorithm takes as input a large corpus and produces as output a set of conflation sets indicating the various inflected and derived forms for each word in the language. As an example, the conflation set of the word "abuse" would contain "abuse", "abused", "abuses", "abusive", "abusively", and so forth. Our algorithm extends earlier approaches to morphology induction by combining various induced information sources: the semantic relatedness of the affixed forms using a Latent Semantic Analysis approach to corpus-based semantics (Schone and Jurafsky, 2000), affix frequency, syntactic context, and transitive closure. Using the hand-labeled CELEX lexicon (Baayen, et al., 1993) as our gold standard, the current version of our algorithm achieves an F-score of 88.1% on the task of identifying conflation sets in English, outperforming earlier algorithms. Our algorithm is also applied to German and Dutch and evaluated on its ability to find prefixes, suffixes, and circumfixes in these languages. To our knowledge, this serves as the first evaluation of complete regular morphological induction of German or Dutch (although researchers such as Nakisa and Hahn (1996) have evaluated induction algorithms on morphological sub-problems in German).

2 Previous Approaches

Previous morphology induction approaches have fallen into three categories. These categories differ depending on whether human input is provided and on whether the goal is to obtain affixes or complete morphological analysis. We here briefly describe work in each category.

2.1 Using a Knowledge Source to Bootstrap

Some researchers begin with some initial human-labeled source from which they induce other morphological components. In particular, Xu and Croft (1998) use word context derived from a corpus to refine Porter stemmer output. Gaussier (1999) induces derivational morphology using an inflectional lexicon which includes part of speech information. Grabar and Zweigenbaum (1999) use the SNOMED corpus of semantically-arranged medical terms to find semantically-motivated morphological relationships. Also, Yarowsky and Wicentowski (2000) obtained outstanding results at in inducing English past tense after beginning with a list of the open class roots in the language, a table of a language's inflectional parts of speech, and the canonical suffixes for each part of speech.

2.2 Affix Inventories

A second, *knowledge-free* category of research has focused on obtaining affix inventories. Brent, et al. (1995) used minimum description length (MDL) to find the most data-compressing suffixes. Kazakov (1997) does something akin to this using MDL as a fitness metric for evolutionary computing. DéJean (1998) uses a strategy similar to that of Harris (1951). He declares that a stem has ended when the number of characters following it exceed some

given threshold and identifies any residual following the stems as suffixes.

2.3 Complete morphological analysis

Due to the existence of morphological ambiguity (such as with the word "caring" whose stem is "care" rather than "car"), finding affixes alone does not constitute a complete morphological analysis. Hence, the last category of research is also knowledge-free but attempts to induce, for each word of a corpus, a complete analysis. Since our approach falls into this category (expanding upon our earlier approach (Schone and Jurafsky, 2000)), we describe work in this area in more detail.

2.3.1 Jacquemin's multiword approach

Jacquemin (1997) deems pairs of word *n*-grams as morphologically related if two words in the first *n*-gram have the same first few letters (or *stem*) as two words in the second *n*-gram *and* if there is a suffix for each stem whose length is less than *k*. He also clusters groups of words having the same kinds of word endings, which gives an added performance boost. He applies his algorithm to a French term list and scores based on sampled, by-hand evaluation.

2.3.2. Goldsmith: EM and MDLs

Goldsmith (1997/2000) tries to automatically sever each word in exactly one place in order to establish a potential set of stems and suffixes. He uses the expectation-maximization algorithm (EM) and MDL as well as some triage procedures to help eliminate inappropriate parses for every word in a corpus. He collects the possible suffixes for each stem and calls these *signatures* which give clues about word classes. With the exceptions of capitalization removal and some word segmentation, Goldsmith's algorithm is otherwise knowledge-free. His algorithm, *Linguistica*, is freely available on the Internet. Goldsmith applies his algorithm to various languages but evaluates in English and French.

2.3.3 Schone and Jurafsky: induced semantics

In our earlier work, we (Schone and Jurafsky (2000)) generated a list of *N* candidate suffixes and used this list to identify word pairs which share the same stem but conclude with distinct candidate suffixes. We then applied Latent Semantic Analysis (Deerwester, et al., 1990) as a method of automatically determining semantic relatedness between word pairs. Using statistics from the semantic relations, we identified those word pairs that have strong semantic correlations as being morphological variants of each other. With the exception of word segmentation, we provided no human information to our system. We applied our system to an English corpus and evaluated by comparing each word's conflation set as produced by our algorithm to those derivable from CELEX.

2.4 Problems with earlier approaches

Most of the existing algorithms described focus on suffixing in inflectional languages (though Jacquemin and DéJean describe work on prefixes). None of these algorithms consider the general conditions of circumfixing or infixing, nor are they applicable to other language types such as agglutinative languages (Sproat, 1992).

Additionally, most approaches have centered around statistics of orthographic properties. We had noted previously (Schone and Jurafsky, 2000), however, that errors can arise from strictly orthographic systems. We had observed in other systems such errors as inappropriate removal of valid affixes ("ally"⇒"all"), failure to resolve morphological ambiguities ("hated"⇒"hat"), and pruning of semi-productive affixes ("dirty"≠"dirt"). Yet we illustrated that induced semantics can help overcome some of these errors.

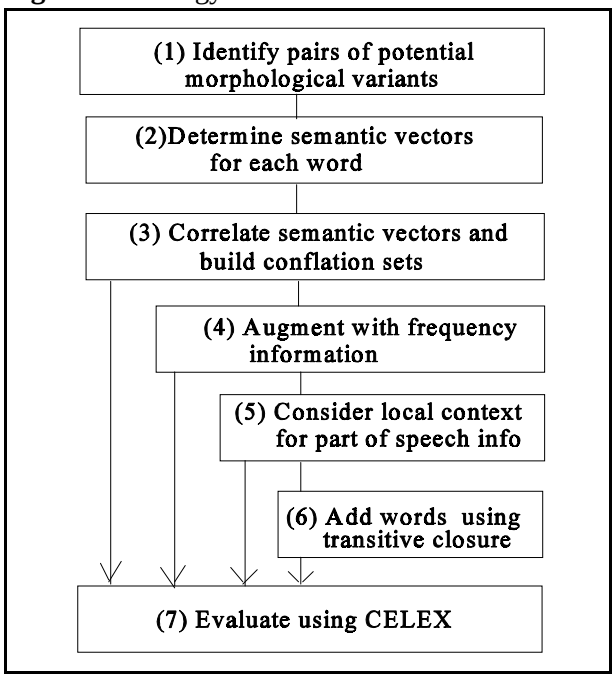
However, we have since observed that induced semantics can give rise to different kinds of problems. For instance, morphological variants may be semantically opaque such that the meaning of one variant cannot be readily determined by the other ("reusability"≠"use"). Additionally, high-frequency function words may be conflated due to having weak semantic information ("as"⇒"a").

Coupling semantic *and* orthographic statistics, as well as introducing induced syntactic information and relational transitivity can help in overcoming these problems. Therefore, we begin with an approach similar to our previous algorithm. Yet we build upon this algorithm in several ways in that we: [1] consider circumfixes, [2] automatically identify capitalizations by treating them similar to prefixes [3] incorporate frequency information, [4] use distributional information to help identify syntactic properties, and [5] use transitive closure to help find variants that may not have been found to be semantically related but which are related to mutual variants. We then apply these strategies to English,

German, and Dutch. We evaluate our algorithm against the human-labeled CELEX lexicon in all three languages and compare our results to those that the Goldsmith and Schone/Jurafsky algorithms would have obtained on our same data. We show how each of our additions result in progressively better overall solutions.

3 Current Approach

Figure 1: Strategy and evaluation



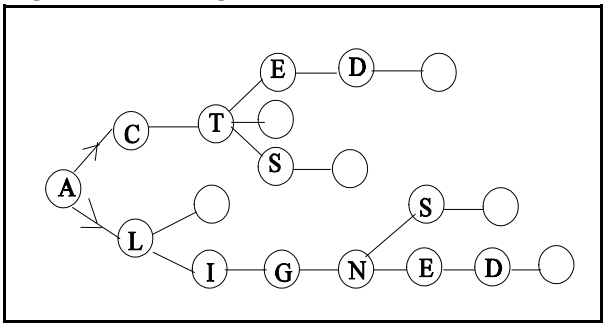
3.1 Finding Candidate Circumfix Pairings

As in our earlier approach (Schone and Jurafsky, 2000), we begin by generating, from an untagged corpus, a list of word pairs that might be morphological variants. Our algorithm has changed somewhat, though, since we previously sought word pairs that vary only by a prefix or a suffix, yet we now wish to generalize to those with circumfixing differences. We use "circumfix" to mean true circumfixes like the German ge-/-t as well as combinations of prefixes and suffixes. It should be mentioned also that we assume the existence of languages having valid circumfixes that are not composed merely of a prefix and a suffix that appear independently elsewhere.

To find potential morphological variants, our first goal is to find word endings which could serve as suffixes. We had shown in our earlier work how one might do this using a character tree, or *trie* (as in Figure 2). Yet using this approach, there may be circumfixes whose endings will be overlooked in the search for suffixes unless we first remove all candidate prefixes. Therefore, we build a lexicon consisting of all words in our corpus and identify all word beginnings with frequencies in excess of some threshold (T_1). We call these *pseudo-prefixes*. We strip all pseudo-prefixes from each word in our lexicon and add the word residuals back into the lexicon as if they were also words. Using this final lexicon, we can now seek for suffixes in a manner equivalent to what we had done before (Schone and Jurafsky, 2000).

To demonstrate how this is done, suppose our initial lexicon $\mathcal{L}$ contained the words "align," "real," "aligns," "realign", "realigned", "react", "reacts," and "reacted." Due to the high frequency occurrence of "re-" suppose it is identified as a pseudo-prefix. If we strip off "re-" from all words, and add all residuals to a trie, the branch of the trie of words beginning with "a" is depicted in Figure 2.

Figure 2: Inserting the residual lexicon into a trie



In our earlier work, we showed that a majority of the regular suffixes in the corpus can be found by identifying trie branches that appear repetitively. By "branch" we mean those places in the trie where some splitting occurs. In the case of Figure 2, for example, the branches NULL (empty circle), "-s" and "-ed" each appear twice. We assemble a list of all trie branches that occur some minimum number of times (T_2) and refer to such as *potential suffixes*.

Given this list, we can now find potential prefixes using a similar strategy. Using our original lexicon, we can now strip off all potential *suffixes* from each word and form a new augmented lexicon. Then, (as we had proposed before) if we reverse the ordering on the words and insert them into a trie, the branches that are formed will be potential prefixes (in reverse order).

Before describing the last steps of this procedure, it is beneficial to define a few terms (some of which appeared in our previous work):

[a] potential circumfix: A pair B/E where B and E occur respectively in potential prefix and suffix lists

[b] pseudo-stem: the residue of a word after its potential circumfix is removed

[c] candidate circumfix: a potential circumfix which appears affixed to at least T_3 pseudo-stems that are shared by other potential circumfixes

[d] rule: a pair of candidate circumfixes sharing at least T_4 pseudo-stems

[e] pair of potential morphological variants (PPMV): two words sharing the same rule but distinct candidate circumfixes

[f] ruleset: the set of all PPMVs for a common rule

Our final goal in this first stage of induction is to find all of the possible rules and their corresponding rulesets. We therefore re-evaluate each word in the original lexicon to identify all potential circumfixes that could have been valid for the word. For example, suppose that the lists of potential suffixes and prefixes contained "-ed" and "re-" respectively. Note also that NULL exists by default in both lists as well. If we consider the word "realigned" from our lexicon $\mathcal{L}$, we would find that its potential circumfixes would be NULL/*ed*, *re*/NULL, and *re/ed* and the corresponding pseudo-stems would be "realign," "aligned," and "align," respectively,

From $\mathcal{L}$, we also note that circumfixes re/ed and NULL/ing share the pseudo-stems "us," "align," and "view" so a rule could be created: re/ed⇒NULL/ing. This means that word pairs such as "reused/using" and "realigned/aligning" would be deemed PPMVs.

Although the choices in T_1 through T_4 is somewhat arbitrary, we chose T_1=T_2=T_3=10 and T_4=3. In English, for example, this yielded 30535 possible rules. Table 1 gives a sampling of these potential rules in each of the three languages in terms of frequency-sorted rank. Notice that several "rules" are quite valid, such as the indication of an English suffix -s. There are also valid circumfixes like the ge-/-t circumfix of German. Capitalization also appears (as a 'prefix'), such as C⇒ c in English, D⇒d in German, and V⇒v in Dutch. Likewise,there are also some rules that may only be true in certain circumstances, such as -d⇒-r in English (such as worked/worker, but certainly not for steed/steer.) However, there are some rules that are wrong: the potential 's-' prefix of English is never valid although word combinations like stick/tick spark/park, and slap/lap happen frequently in English. Incorporating semantics can help determine the validity of each rule.

Table 1: Outputs of the trie stage: potential rules

Rank	ENGLISH	GERMAN	DUTCH
1	-s⇒ ∅	-n⇒ ∅	-en⇒ ∅
2	-ed⇒ -ing	-en⇒ ∅	-e⇒ ∅
4	-ing⇒ ∅	-s⇒ ∅	-n⇒ ∅
8	-ly⇒ ∅	-en⇒ -t	de-⇒ ∅
12	C-⇒ c-	-en⇒ -te	-er⇒ ∅
16	re-⇒ ∅	1-⇒ ∅	-r⇒ ∅
20	-ers⇒ -ing	er-⇒ ∅	V-⇒ v-
24	1-⇒ ∅	1-⇒ 2-	-ingen ⇒ -e
28	-d⇒ -r	ge-/-t ⇒ -en	ge-⇒ -e
32	s-⇒ ∅	D-⇒ d-	-n⇒ -rs

3.2 Computing Semantics

Deerwester, et al. (1990) introduced an algorithm called Latent Semantic Analysis (LSA) which showed that valid semantic relationships between words and documents in a corpus can be induced with virtually no human intervention. To do this, one typically begins by applying singular value decomposition (SVD) to a matrix, *M*, whose entries *M*(i,j) contains the frequency of word *i* as seen in document *j* of the corpus. The SVD decomposes *M* into the product of three matrices, *U*, *D*, and V^T such that *U* and V^T are orthogonal matrices and *D* is a diagonal matrix whose entries are the singular values of *M*. The LSA approach then zeros out all but the top *k* singular values of the SVD, which has the effect of projecting vectors into an optimal *k*-dimensional subspace. This methodology is well-described in the literature (Landauer, et al., 1998; Manning and Schütze, 1999).

In order to obtain semantic representations of each word, we apply our previous strategy (Schone and Jurafsky (2000)). Rather than using a term-document matrix, we had followed an approach akin to that of Schütze (1993), who performed SVD on a *N*x*2N* term-term matrix. The *N* here represents the *N*-1 most-frequent words as well as a glob position to account for all other words not in the top *N*-1. The matrix is structured such that for a given word *w*'s row, the first *N* columns denote words that

precede *w* by up to 50 words, and the second *N* columns represent those words that follow by up to 50 words. Since SVDs are more designed to work with normally-distributed data (Manning and Schütze, 1999, p. 565), we fill each entry with a normalized count (or Z-score) rather than straight frequency. We then compute the SVD and keep the top 300 singular values to form semantic vectors for each word. Word *w* would be assigned the semantic vector $\mathbf{\Omega_W}=U_w D_k$, where U_w represents the row of *U* corresponding to *w* and D_k indicates that only the top *k* diagonal entries of *D* have been preserved.

As a last comment, one would like to be able to obtain a separate semantic vector for every word (not just those in the top *N*). SVD computations can be expensive and impractical for large values of *N*. Yet due to the fact that *U* and V^T are orthogonal matrices, we can start with a matrix of reasonable-sized *N* and "fold in" the remaining terms, which is the approach we have followed. For details about folding in terms, the reader is referred to Manning and Schütze (1999, p. 563).

3.3 Correlating Semantic Vectors

To correlate these semantic vectors, we use normalized cosine scores (NCSs) as we had illustrated before (Schone and Jurafsky (2000)). The normalized cosine score between two words w_1 and w_2 is determined by first computing cosine values between each word's semantic vector and 200 other randomly selected semantic vectors. This provides a mean (μ) and variance (σ^2) of correlation for each word. The NCS is given to be

$$NCS(w_1,w_2) = \min_{k\in(1,2)} \frac{\cos(\Omega_{w1},\Omega_{w2})-\mu_k}{\sigma_k} \qquad (1)$$

We had previously illustrated NCS values on various PPMVs and showed that this type of score seems to be appropriately identifying semantic relationships. (For example, the PPMVs of car/cars and ally/allies had NCS values of 5.6 and 6.5 respectively, whereas car/cares and ally/all had scored only -0.14 and -1.3.) Further, we showed that by performing this normalizing process, one can estimate the probability that an NCS is random or not. We expect that random NCSs will be approximately normally distributed according to $N(0,1)$. We can also estimate the distribution $N(\mu_T,\sigma_T^2)$ of true correlations and number of terms in that distribution (n_T). If we define a function

$$\Phi_{NCS}(\mu,\sigma) = \int_{NCS}^{\infty} \exp[-((x-\mu)/\sigma)^2]\,dx$$

then, if there were n_R items in the ruleset, the probability that a NCS is non-random is

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

We define $Pr_{sem}(w_1\Rightarrow w_2)=Pr(NCS(w_1,w_2))$. We choose to accept as valid relationships only those PPMVs with $Pr_{sem}\geq T_5$, where T_5 is an acceptance threshold. We showed in our earlier work that T_5=85% affords high overall precision while still identifying most valid morphological relationships.

3.4 Augmenting with Affix Frequencies

The first major change to our previous algorithm is an attempt to overcome some of the weaknesses of purely semantic-based morphology induction by incorporating information about affix frequencies. As validated by Kazakov (1997), high frequency word endings and beginnings in inflectional languages are very likely to be legitimate affixes. In English, for example, the highest frequency rule is -s⇒∅. CELEX suggests that 99.7% of our PPMVs for this rule would be true. However, since the purely semantic-based approach tends to select only relationships with contextually similar meanings, only 92% of the PPMVs are retained. This suggests that one might improve the analysis by supplementing semantic probabilities with orthographic-based probabilities (Pr_{orth}).

Our approach to obtaining Pr_{orth} is motivated by an appeal to minimum edit distance (MED). MED has been applied to the morphology induction problem by other researchers (such as Yarowsky and Wicentowski, 2000). MED determines the minimum-weighted set of insertions, substitutions, and deletions required to transform one word into another. For example, only a single deletion is required to transform "rates" into "rate" whereas two substitutions and an insertion are required to transform it into "rating." Effectively, if Cost(•) is transforming cost, Cost(rates⇒rate) = Cost(s⇒∅) whereas Cost(rates⇒rating)=Cost(es⇒ing). More generally, suppose word X has circumfix $C_1=B_1/E_1$ and pseudo-stem -S-, and word Y has circumfix $C_2=B_2/E_2$ also with pseudo-stem -S-. Then, $Cost(X\Rightarrow Y)=Cost(B_1SE_1\Rightarrow B_2SE_2)=Cost(C_1\Rightarrow C_2)$. Since we are free to choose whatever cost function we desire, we can equally choose one whose range

lies in the interval of [0,1]. Hence, we can assign $Pr_{orth}(X \Rightarrow Y) = \text{Cost}(X \Rightarrow Y)$. This calculation implies that the *orthographic* probability that X and Y are morphological variants is simply the cost of transforming C_1 into C_2.

The only question remaining is how to determine $\text{Cost}(C_1 \Rightarrow C_2)$. This cost should depend on a number of factors: the frequency of the rule $f(C_1 \Rightarrow C_2)$, the reliability of the metric in comparison to that of semantics (α, where $\alpha \in [0,1]$), and the frequencies of other rules involving C_1 and C_2. We define the orthographic probability of validity as

$$Cost(C_1 \Rightarrow C_2) = \frac{2\,\alpha\, f(C_1 \Rightarrow C_2)}{\max_{\forall Z} f(C_1 \Rightarrow Z) + \max_{\forall W} f(W \Rightarrow C_2)}$$

We suppose that orthographic information is less reliable than semantic information, so we arbitrarily set α=0.5. Now since $Pr_{orth}(X \Rightarrow Y) = \text{Cost}(C_1 \Rightarrow C_2)$, we can readily combine it with Pr_{sem} if we assume independence using the "noisy or" formulation:

$$Pr_{s\text{-}o}(\text{valid}) = Pr_{sem} + Pr_{orth} - (Pr_{sem}\, Pr_{orth}). \quad (2)$$

By using this formula, we obtain 3% (absolute) more of the correct PPMVs than semantics alone had provided for the -s⇒∅ rule and, as will be shown later, gives reasonable improvements overall.

3.5 Local Syntactic Context

Since a primary role of morphology — inflectional morphology in particular — is to convey *syntactic* information, there is no guarantee that two words that are morphological variants need to share similar *semantic* properties. This suggests that performance could improve if the induction process took advantage of local, syntactic contexts around words in addition to the more global, large-window contexts used in semantic processing.

Table 2: Sample probabilities for "-s⇒∅"

Word+s	Word	Pr	Word+s	Word	Pr
agendas	agenda	.968	legends	legend	.981
ideas	idea	.974	militias	militia	1.00
pleas	plea	1.00	guerrillas	guerrilla	1.00
seas	sea	1.00	formulas	formula	1.00
areas	area	1.00	railroads	railroad	1.00
Areas	Area	.721	pads	pad	.731
Vegas	Vega	.641	feeds	feed	.543

Consider Table 2 which is a sample of PPMVs from the ruleset for "-s⇒∅" along with their probabilities of validity. A validity threshold (T_5) of 85% would mean that the four bottom PPMVs would be deemed invalid. Yet if we find that the local contexts of these low-scoring word pairs match the contexts of other PPMVs having high scores (i.e., those whose scores exceed T_5), then their probabilities of validity should increase. If we could compute a syntax-based probability for these words, namely Pr_{syntax}, then assuming independence we would have:

$$Pr(\text{valid}) = Pr_{s\text{-}o} + Pr_{syntax} - (Pr_{s\text{-}o}\, Pr_{syntax})$$

Figure 3 describes the pseudo-code for an algorithm to compute Pr_{syntax}. Essentially, the algorithm has two major components. First, for left (L) and right-hand (R) sides of each valid PPMV of a given ruleset, try to find a collection of words from the corpus that are collocated with L and R but which occur statistically too many or too few times in these collocations. Such word sets form *signatures*. Then, determine similar signatures for a randomly-chosen set of words from the corpus as well as for each of the PPMVs of the ruleset that are not yet validated. Lastly, compute the NCS and their corresponding probabilities (see equation 1) between the ruleset's signatures and those of the to-be-validated PPMVs to see if they can be validated.

Table 3 gives an example of the kinds of contextual words one might expect for the "-s⇒∅" rule. In fact, the syntactic signature for "-s⇒∅" does indeed include such words as *are*, *other*, *these*, *two*, *were*, and *have* as indicators of words that occur on the left-hand side of the ruleset, and *a*, *an*, *this*, *is*, *has*, and *A* as indicators of the right-hand side. These terms help distinguish plurals from singulars.

Table 3: Examples of "-s⇒∅" contexts

Context for *L*		Context for *R*	
agendas *are*	seas *were*	*a* legend	*this* formula
two red pads	pleas *have*	militia *is*	*an* area
these ideas	*other* areas	railroad *has*	*A* guerrilla

There is an added benefit from following this approach: it can also be used to find rules that, though different, seem to convey similar information . Table 4 illustrates a number of such agreements. We have yet to take advantage of this feature, but it clearly could be of use for part-of-speech induction.

Figure 3: Pseudo-code to find $Probability_{syntax}$

```
procedure SyntaxProb(ruleset,corpus)
   leftSig  ⇐GetSignature(ruleset,corpus,left)
   rightSig⇐GetSignature(ruleset,corpus,right)
   Ω_ruleset   ⇐Concatenate(leftSig, rightSig)
   (μ_ruleset,σ_ruleset)⇐ComparetoRandom(Ω_ruleset)
   foreach PPMV in ruleset
     if  (Pr_S-O(PPMV) ≥ T_5 )  continue
     wLSig⇐GetSignature(PPMV,corpus,left)
     wRSig⇐GetSignature(PPMV,corpus,right)
     Ω_PPMV ⇐Concatenate(wLSig, wRSig)
     (μ_PPMV,σ_PPMV)⇐ComparetoRandom(Ω_PPMV)
     prob[PPMV]⇐Pr(NCS(PPMV,ruleset))
end procedure

function GetSignature(ruleset,corpus,side)
   foreach PPMV in ruleset
     if  (Pr_S-O(PPMV) < T_5 )  continue
     if  (side=left) X ⇐ LeftWordOf(PPMV)
     else  X ⇐ RightWordOf(PPMV)
     CountNeighbors(corpus,colloc,X)
   colloc  ⇐SortWordsByFreq(colloc)
   for i ⇐ 1 to 100 signature[i]⇐colloc[i]
   return signature
end function

procedure CountNeighbors(corpus,colloc,X)
  foreach W in Corpus
     push(lexicon,W)
     if (PositionalDistanceBetween(X,W)≤2)
       count[W] ⇐ count[W]+1
   foreach W in lexicon
     if ( Zscore(count[W])≥ 3.0   or
          Zscore(count[W])≤ -3.0)
        colloc[W]⇐colloc[W]+1
end procedure
```

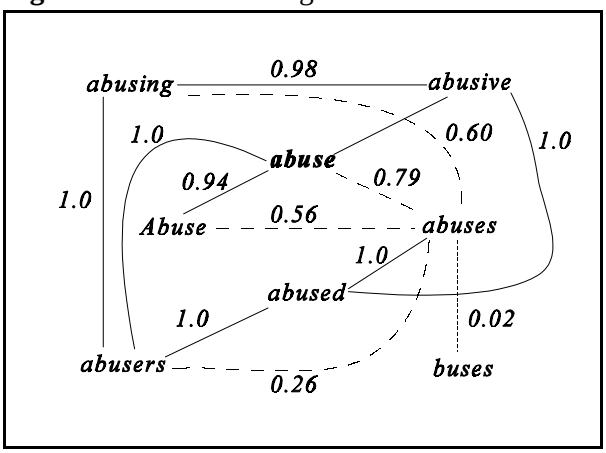
Table 4: Relations amongst rules

Rule	Relative	Cos	Rule	Relative	Cos
-s⇒∅	-ies⇒y	83.8	-ed⇒∅	-d⇒∅	95.5
-s⇒∅	-es⇒∅	79.5	-ing⇒∅	-e⇒∅	94.3
-ed⇒∅	-ied⇒y	81.9	-ing⇒∅	-ting⇒∅	70.7

3.6 Branching Transitive Closure

Despite the semantic, orthographic, and syntactic components of the algorithm, there are still valid PPMVs, (X⇒Y), that may seem unrelated due to corpus choice or weak distributional properties. However, X and Y may appear as members of other *valid* PPMVs such as (X⇒Z) and (Z⇒Y) containing variants (Z, in this case) which are either semantically or syntactically related to both of the other words. Figure 4 demonstrates this property in greater detail. The words conveyed in Figure 4 are all words from the corpus that have potential relationships between variants of the word "abuse." Links between two words, such as "abuse" and "Abuse," are labeled with a weight which is the semantic correlation derived by LSA. Solid lines represent valid relationships with $Pr_{sem} \geq 0.85$ and dashed lines indicate relationships with lower-than-threshold scores. The absence of a link suggests that either the potential relationship was never identified or discarded at an earlier stage. Self loops are assumed for each node since clearly each word should be related morphologically to itself. Since there are seven words that are valid morphological relationships of "abuse," we would like to see a complete graph containing 21 solid edges. Yet, only eight connections can be found by semantics alone (Abuse⇒abuse, abusers⇒abusing, etc.).

Figure 4: Semantic strengths



However, note that there is a path that can be followed along solid edges from every correct word to every other correct variant. This suggests that taking into consideration link transitivity (i.e., if $X \Rightarrow Y_1$, $Y_1 \Rightarrow Y_2$, $Y_2 \Rightarrow Y_3$,... and $Y_t \Rightarrow Z$, then $X \Rightarrow Z$) may drastically reduce the number of deletions.

There are two caveats that need to be considered for transitivity to be properly pursued. The first caveat: if no rule exists that would transform X into Z, we will assume that despite the fact that there may be a probabilistic path between the two, we

will disregard such a path. The second caveat is that we will say that paths can only consist of solid edges, namely each $Pr(Y_i \Rightarrow Y_{i+1})$ on every path must exceed the specified threshold.

Given these constraints, suppose now there is a transitive relation from X to Z by way of some intermediate path $\pi_i=\{Y_1, Y_2, \ldots Y_t\}$. That is, assume there is a path $X \Rightarrow Y_1$, $Y_1 \Rightarrow Y_2, \ldots, Y_t \Rightarrow Z$. Suppose also that the probabilities of these relationships are respectively $p_0, p_1, p_2, \ldots, p_t$. If β is a decay factor in the unit interval accounting for the number of link separations, then we will say that the $Pr(X \Rightarrow Z)$ along path π_i has probability $r_{\pi i} = \beta^t \prod_{j=0}^{t} p_j$ We combine the probabilities of all independent paths between X and Z according to Figure 5:

Figure 5: Pseudocode for Branching Probability

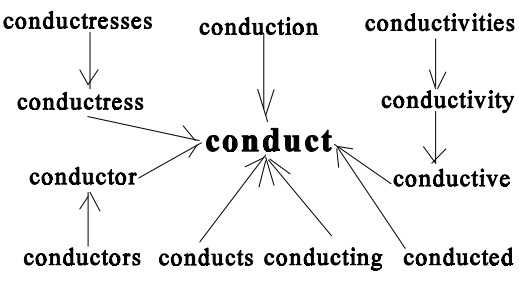
```
function BranchProbBetween(X,Z)
prob⇐0
foreach independent path π_j
   prob ⇐ prob+Pr_πj(X⇒Z) - (prob*Pr_πj(X⇒Z) )
return prob
```

If the returned probability exceeds T_5, we declare X and Z to be morphological variants of each other.

4 Evaluation

We compare this improved algorithm to our former algorithm (Schone and Jurafsky (2000)) as well as to Goldsmith's *Linguistica* (2000). We use as input to our system 6.7 million words of English newswire, 2.3 million of German, and 6.7 million of Dutch. Our gold standards are the hand-tagged morphologically-analyzed CELEX lexicon in each of these languages (Baayen, et al., 1993). We apply the algorithms only to those words of our corpora with frequencies of 10 or more. Obviously this cut-off slightly limits the generality of our results, but it also greatly decreases processing time for all of the algorithms we test against. Furthermore, since CELEX has limited coverage, many of these lower-frequency words could not be scored anyway. This cut-off also helps each of the algorithms to obtain stronger statistical information on the words they do process which means that any observed failures cannot be attributed to weak statistics.

Figure 6: Morphologic relations of "conduct"



Morphological relationships can be represented as directed graphs. Figure 6, for instance, illustrates the directed graph, according to CELEX, of words associated with "conduct." We will call the words of such a directed graph the *conflation set* for any of the words in the graph. Due to the difficulty in developing a scoring algorithm to compare directed graphs, we will follow our earlier approach and only compare induced conflation sets to those of CELEX. To evaluate, we compute the number of correct (***C***), inserted (***I***), and deleted (***D***) words each algorithm predicts for each hypothesized conflation set. If X_w represents word *w*'s conflation set according to an algorithm, and if Y_w represents its CELEX-based conflation set, then,

$$\boldsymbol{C} = \Sigma\forall w(|X_w \cap Y_w|/|Y_w|),$$
$$\boldsymbol{D} = \Sigma\forall w(|Y_w - (X_w \cap Y_w)|/|Y_w|), \text{ and}$$
$$\boldsymbol{I} = \Sigma\forall w\,(|X_w - (X_w \cap Y_w)|/|Y_w|),$$

In making these computations, we disregard any CELEX words absent from our data set and vice versa. Most capital words are not in CELEX so this process also discards them. Hence, we also make an augmented CELEX to incorporate capitalized forms.

Table 5 uses the above scoring mechanism to compare the F-Scores (product of precision and recall divided by average of the two) of our system at a cutoff threshold of 85% to those of our earlier algorithm ("S/J2000") at the same threshold; Goldsmith; and a baseline system which performs no analysis (claiming that for any word, its conflation set only consists of itself). The "S" and "C" columns respectively indicate performance of systems when scoring for suffixing and circumfixing (using the unaugmented CELEX). The "A" column shows circumfixing performance using the augmented CELEX. Space limitations required that we illustrate "A" scores for one language only, but performance in the other two language is similarly degraded. Boxes are shaded out for algorithms not designed to produce circumfixes.

Note that each of our additions resulted in an overall improvement which held true across each of

the three languages. Furthermore, using ten-fold cross validation on the English data, we find that F-score differences of the S column are each statistically significant at least at the 95% level.

Table 5: Computation of F-Scores

Algorithms	English			German		Dutch	
	S	C	A	S	C	S	C
None	62.8	59.9	51.7	75.8	63.0	74.2	70.0
Goldsmith	81.8			84.0		75.8	
S/J2000	85.2			88.3		82.2	
+orthogrph	85.7	82.2	76.9	89.3	76.1	84.5	78.9
+ syntax	87.5	84.0	79.0	91.6	78.2	85.6	79.4
+ transitive	**88.1**	84.5	79.7	**92.3**	78.9	**85.8**	79.6

5 Conclusions

We have illustrated three extensions to our earlier morphology induction work (Schone and Jurafsky (2000)). In addition to induced semantics, we incorporated induced orthographic, syntactic, and transitive information resulting in almost a 20% relative reduction in overall induction error. We have also extended the work by illustrating performance in German and Dutch where, to our knowledge, complete morphology induction performance measures have not previously been obtained. Lastly, we showed a mechanism whereby circumfixes as well as combinations of prefixing and suffixing can be induced in lieu of the suffix-only strategies prevailing in most previous research.

For the future, we expect improvements could be derived by coupling this work, which focuses primarily on inducing regular morphology, with that of Yarowsky and Wicentowski (2000), who assume some information about regular morphology in order to induce irregular morphology. We also believe that some findings of this work can benefit other areas of linguistic induction, such as part of speech.

Acknowledgments

The authors wish to thank the anonymous reviewers for their thorough review and insightful comments.

References

Baayen, R.H., R. Piepenbrock, and H. van Rijn. (1993) The CELEX lexical database (CD-ROM), LDC, Univ. of Pennsylvania, Philadelphia, PA.

Brent, M., S. K. Murthy, A. Lundberg. (1995). Discovering morphemic suffixes: A case study in MDL induction. Proc. Of 5th Int'l Workshop on Artificial Intelligence and Statistics

DéJean, H. (1998) Morphemes as necessary concepts for structures: Discovery from untagged corpora. *Workshop on paradigms and Grounding in Natural Language Learning*, pp. 295-299.Adelaide, Australia

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 of Information Science*, Vol. 41, pp.391-407.

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

Goldsmith, J. (1997/2000) Unsupervised learning of the morphology of a natural language. Univ. of Chicago. http://humanities.uchicago.edu/faculty/goldsmith.

Grabar, N. and P. Zweigenbaum. (1999) Acquisition automatique de connaissances morphologiques sur le vocabulaire médical, *TALN*, Cargése, France.

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

Jacquemin, C. (1997) Guessing morphology from terms and corpora. *SIGIR'97*, pp. 156-167, Philadelphia, PA.

Kazakov, D. (1997) Unsupervised learning of naïve morphology with genetic algorithms. In W. Daelemans, et al., eds., *ECML/Mlnet Workshop on Empirical Learning of Natural Language Processing Tasks*, Prague, pp. 105-111.

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

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

Nakisa, R.C., U.Hahn. (1996) Where defaults don' t help: the case of the German plural system. *Proc. of the 18th Conference of the Cognitive Science Society.*

Schone, P. and D. Jurafsky. (2000) Knowledge-free induction of morphology using latent semantic analysis. *Proc. of the Computational Natural Language Learning Conference*, Lisbon, pp. 67-72.

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.

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

Xu, J., B.W. Croft. (1998) Corpus-based stemming using co-occurrence of word variants. *ACM Transactions on Information Systems*, 16 (1), pp. 61-81.

Yarowsky, D. and R. Wicentowski. (2000) Minimally supervised morphological analysis by multimodal alignment. *Proc. of the ACL 2000*, Hong Kong.