2004

Unsupervised Induction of Natural Language Morphology Inflection Classes

Christian Monson
Carnegie Mellon University

Alon Lavie
Carnegie Mellon University

Jaime G. Carbonell
Carnegie Mellon University

Lori Levin
Carnegie Mellon University

Unsupervised Induction of Natural Language Morphology Inflection Classes

Christian Monson, Alon Lavie, Jaime Carbonell, Lori Levin
Language Technologies Institute
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, USA 15213
{cmonson, alavie+, jgc+, lsl+}@cs.cmu.edu

Abstract

We propose a novel language-independent framework for inducing a collection of morphological inflection classes from a monolingual corpus of full form words. Our approach involves two main stages. In the first stage, we generate a large data structure of candidate inflection classes and their interrelationships. In the second stage, search and filtering techniques are applied to this data structure, to identify a select collection of "true" inflection classes of the language. We describe the basic methodology involved in both stages of our approach and present an evaluation of our baseline techniques applied to induction of major inflection classes of Spanish. The preliminary results on an initial training corpus already surpass an F_1 of 0.5 against ideal Spanish inflectional morphology classes.

1 Introduction

Many natural language processing tasks, such as morphological analysis and parsing, have mature solutions when applied to resource-rich European and Asian languages. Addressing these same tasks in less studied low-density languages, however, poses exciting challenges.

These languages have limited available resources: with perhaps a few million speakers there is likely no native speaker linguist and frequently there is little electronic text readily available. To compound the difficulties, while low-density languages abound, comparatively little financial resources are available to address their challenges. These considerations suggest developing systems to automatically induce solutions for NLP tasks in new languages.

The AVENUE project (Lavie et al, 2003; Carbonell et al., 2002; Probst et al., 2002) at Carnegie Mellon University seeks to apply automatic induction methods to develop rule-based machine translation systems between pairs of languages where one of the languages is low-density and the other is resource-rich. We are currently pursuing MT systems with Mapudungun, an indigenous language spoken by 900,000 people in southern Chile and Argentina, and Aymara, spoken by 3 million people in Bolivia, Peru, and northern Chile, as low-density languages and Spanish the resource rich language.

A vital first step in a rule-based machine translation system is morphological analysis. This paper outlines a framework for automatic natural language morphology induction inspired by the traditional and linguistic concept of inflection classes. Additional details concerning the candidate inflection class framework can be found in Monson (2004). This paper then goes on to describe one implemented search strategy within this framework, presenting both a simple summary of results and an in depth error analysis.

While the intent of this research direction is to define techniques applicable to low-density languages, this paper employs English to illustrate the main conjectures and Spanish, a language with a reasonably complex morphological system, for quantitative analysis. All experiments detailed in this paper are over a Spanish newswire corpus of 40,011 tokens and 6,975 types.

2 Previous Work

It is possible to organize much of the recent work on unsupervised morphology induction by considering the bias each approach has toward discovering morphologically related words that are also orthographically similar.

At one end of the spectrum is the work of Yarowsky et al. (2001), who derive a morphological analyzer for a language, L, by projecting the morphological analysis of a resource-rich language onto L through a clever application of statistical machine translation style word alignment probabilities. The word alignments are trained over a sentence aligned parallel bilingual text for the language pair. While the probabilistic model they use to generalize their initial system contains a bias toward orthographic similarity, the unembellished algorithm contains no assumptions on the orthographic shape of related word forms.

Next along the spectrum of orthographic similar-

Verb Paradigm	Inflection Classes		
	A	**B**	**C**
Basic	blame roam solve	show sow saw	sing ring
3rd Person Singular Non-past	-/z/ blames roams solves	-/z/ shows sows saws	-/z/ sings rings
Past	-/d/ blamed roamed solved	-/d/ showed sowed sawed	V→ /eɪ/ sang rang
Perfective or *Passive*	-/d/ blamed roamed solved	-/n/ shown sown sawn	V→ /ʌ/ sung rung
Progressive	-/iŋ/ blaming roaming solving	-/iŋ/ showing sowing sawing	-/iŋ/ singing ringing

Table 1: A few inflection classes of the English verb paradigm

ity bias is the work of Schone and Jurafsky (2000), who first acquire a list of pairs of potential morphological variants (PPMV's) using an orthographic similarity technique due to Gaussier (1999), in which pairs of words from a corpus vocabulary with the same initial string are identified. They then apply latent semantic analysis (LSA) to score each PPMV with a semantic distance. Pairs measuring a small distance, those whose potential variants tend to occur where a neighborhood of the nearest hundred words contains similar counts of individual high-frequency forms, are then proposed as true morphological variants of one anther. In later work, Schone and Jurafsky (2001) extend their technique to identify not only suffixes but also prefixes and circumfixes by building both forward and backward tries over a corpus.

Goldsmith (2001), by searching over a space of morphology models limited to substitution of suffixes, ties morphology yet closer to orthography. Segmenting word forms in a corpus, Goldsmith creates an inventory of stems and suffixes. Suffixes which can interchangeably concatenate onto a set of stems form a signature. After defining the space of signatures, Goldsmith searches for that choice of word segmentations resulting in a minimum description length local optimum.

Finally, the work of Harris (1955; 1967), and later Hafer and Weiss (1974), has direct bearing on the approach taken in this paper. Couched in modern terms, their work involves first building tries over a corpus vocabulary, and then selecting, as morpheme boundaries, those character boundaries with high branching count in the tries.

The work in this paper also has a strong bias toward discovering morphologically related words that share a similar orthography. In particular, the morphology model we use is, akin to Goldsmith, limited to suffix substitution. The novel proposal we bring to the table, however, is a formalization of the full search space of all candidate inflection classes. With this bulwark in place, defining search strategies for morpheme discovery becomes a natural and straightforward activity.

3 Inflection Classes as Motivation

When learning the morphology of a foreign language, it is common for a student to study tables of inflection classes. In Spanish, for example, a regular verb belongs to one of three inflection classes—verbs that take the *-ar* infinitive suffix inflect for various syntactic features with one set of suffixes, verbs that take the *-er* infinitive suffix realize the same set of syntactic features with a second set of suffixes, while *-ir* verbs take yet a third set.

Carstairs-McCarthy formalizes the concept of an inflection class in chapter 16 of The Handbook of Morphology (1998). In his terminology, a language with inflectional morphology contains lexemes which occur in a variety of word forms. Each word form carries two pieces of information:

1) Lexical content and
2) Morphosyntactic properties.

For example, the English word form *gave* expresses the lexeme GIVE plus the morphosyntactic property *Past*, while *gives* expresses GIVE plus the properties *3rd Person*, *Singular*, and *Non-Past*.

A set of morphosyntactic properties realized with a single word form is defined to be a *cell*, while a *paradigm* is a set of cells exactly expressed by the word forms of some lexeme.

A particular natural language may have many paradigms. In English, a language with very little inflectional morphology, there are at least two paradigms, a noun paradigm consisting of two cells, *Singular* and *Plural*, and a paradigm for verbs, consisting of the five cells given (with one choice of naming convention) as the first column of Table 1.

Lexemes that belong to the same paradigm may still differ in their morphophonemic realizations of various cells in that paradigm—each paradigm may have several associated *inflection classes* which specify, for the lexemes belonging to that inflection class, the surface instantiation for each cell of the paradigm.

Three of the inflection classes within the English verb paradigm are found in Table 1 under the columns labeled A through C. Each inflection class

Vocabulary:	blame		solve			
	blames	roams	solves			
	blamed	roamed				
		roaming	solving			

lame.lames.lamed *b*	**ame.ames.amed** *bl*	**me.mes.med** *bla*	**e.es.ed** *blam*	**Ø.s.d** *blame*	**oams.oamed.oaming** *r*	
lame.lames *b*	**ame.ames** *bl*	**me.mes** *bla*	**e.es** *blam.solv*	**Ø.s** *blame.solve*	**olve.olves.olving** *s*	
lame.lamed *b*	**ame.amed** *bl*	**me.med** *bla*	**e.ed** *blam*	**Ø.d** *blame*	...	
lames.lamed *b*	**ames.amed** *bl*	**mes.med** *bla*	**es.ed** *blam*	**s.d** *blame*	**s.ed.ing** *roam*	**e.es.ing** *solv*
lame *b*	**ame** *bl*	**me** *bla*	**e** *blam.solv*		**s.ed** *roam*	**e.ing** *solv*
lames *b*	**ames** *bl*	**mes** *bla*	**es** *blam.solv*	**s** *blame.roam.solve*	**s.ing** *roam*	**es.ing** *solv*
lamed *b*	**amed** *bl.ro*	**med** *bla.roa*	**ed** *blam.roam*	**d** *blame.roame*	**ed.ing** *roam*	**ng** *roami.solvi*
Ø *blame.blames.blamed.roams.roamed.roaming.solve.solves.solving*					**ing** *roam.solv*	**g** *roamin.solvin*

Table 2: Some of the CIC's, arranged in a systematic but arbitrary order, derived from a toy vocabulary. Each entry is specified as a period delimited sequence of c-suffixes in **bold** above a period delimited sequence of adherent c-stems in *italics*

column consists of entries corresponding to the cells of the verb paradigm. Each entry contains an informal notation for the morphophonemic process which the inflection class applies to the basic form of a lexeme and examples of word forms filling the corresponding paradigm cell.

Inflection class A is one of the largest and most productive verb inflection classes in English, inflection class B contains the Perfective/Passive suffix -/n/, and C is a small "irregular" inflection class of strong verbs.

The task our morphology induction system engages is exactly the discovery of the inflection classes of a natural language. Unlike the analysis in Table 1, however, the rest of this paper treats word forms as simply strings of characters as opposed to strings of phonemes.

4 Empirical Inflection Classes

There are two stages in our approach to unsupervised morphology induction. First, we define a search space over a set of candidate inflection classes, and second, we search this space for those candidates most likely to be part of a true inflection class in the language. In both stages of our approach we intentionally exploit the fact that suffixes belonging to the same natural language inflection class frequently occur interchangeably on the same stems.

4.1 Candidate Inflection Class Search Space

To define a search space wherein we hope to identify inflection classes of a natural language, our algorithm accepts as input a monolingual corpus for that language and proposes candidate morpheme boundaries at every character boundary in every word form in the corpus vocabulary. We call each string before a candidate morpheme boundary a *candidate stem* or *c-stem*, and each string after a boundary a *c-suffix*. We define a *candidate inflection class* (CIC) to be a set of c-suffixes for which there exists at least one c-stem, t, such that each c-suffix in the CIC concatenated to t produces a word form in the vocabulary. For convenience, let the set of c-stems which generate a CIC, C, be called the *adherent c-stems* of C; let the number of adherent c-stems of C be C's *adherent size*; and let the size of the set of c-suffixes in C be the *level* of C. We denote a CIC in this paper by a period delimited sequence of c-suffixes.

While CIC's effectively model suffix substitution on bound stems, we would also like to model suffix concatenation onto free stems. To this end, the set of candidate morpheme boundaries our algorithm proposes include those boundaries after the final character in each word form. In this paper we assume a suffix, which we denote as Ø, follows all word form final boundaries. A CIC contains the Ø c-suffix when each c-stem in the CIC can occur, not only bound to other c-suffixes in the CIC, but also as a free stem. For generality, the boundary before the first character of each word form is also a candidate morpheme boundary.

Table 2 illustrates the type of CIC's produced by our algorithm. The CIC's in this table, arranged in a systematic but arbitrary order, are each derived

from one or more forms in a small vocabulary consisting of a subset of the word forms found under inflection class A from Table 1. Proposing, as our procedure does, morpheme boundaries at every character boundary in every word form necessarily produces many ridiculous CIC's, such as **ame.ames.amed**, from the forms *blame*, *blames*, and *blamed* and the c-stem *bl*. Dispersed among the incorrect CIC's generated by our algorithm, however, are also CIC's that seem very reasonable, such as **Ø.s**, from the c-stems *blame* and *tease*.

Note that where Table 1 lists all the surface forms of the three lexemes BLAME, ROAM, and SOLVE, the vocabulary of Table 2 mimics the vocabulary of a text corpus from a highly inflected language where we expect few, if any, lexemes to occur in the complete set of possible surface forms. Specifically, the vocabulary of Table 2 lacks the surface form *blaming* of the lexeme BLAME, *solved* of the lexeme SOLVE, and the root form *roam* of the lexeme ROAM. Hence, while the reasonable CIC **Ø.s** arises from the pairs of surface forms (*blame, blames*) and (*solve, solves*), there is no way for the form *roams* to contribute to the **Ø.s** CIC because the surface form *roam* is missing from this vocabulary. In other words, we lack evidence for a Ø suffix on the c-stem *roam*. Also notice that, as a result of English spelling rules, the CIC **s.ed** generated from the pair of surface forms (*roams, roamed*) is separate from each of the CIC's **s.d** and **es.ed** generated from the pair of surface forms (*blames, blamed*).

Looking at Table 2, it is clear there is structure among the CIC's. In particular, at least two types of relations hold between CIC's. First, hierarchically, the c-suffixes of one CIC may be a superset of the c-suffixes of another CIC. For example the c-suffixes in the CIC **e.es.ed** are a superset of the c-suffixes in the CIC **e.ed**. Second, cutting across this hierarchical structure there is structure between CIC's which propose different morpheme boundaries within the same word forms. Compare the CIC's **me.mes.med** and **e.es.ed**; each is derived from exactly the triple of word forms *blame, blames,* and *blamed,* but differ in the placement of the hypothesized morpheme boundary.

Taken together the hierarchical c-suffix set inclusion relations and the morpheme boundary relations impose a lattice structure on the space of CIC's. Figure 1 diagrams the CIC lattice over an interesting subset of the columns of Table 2. Hierarchical links, represented by solid lines, connect any given CIC often to more than one parent and more than one child. The empty CIC (not pictured in Figure 1) can be considered the child of all level one CIC's (including the **Ø** CIC), but there is no universal parent of all top level CIC's. Moving up

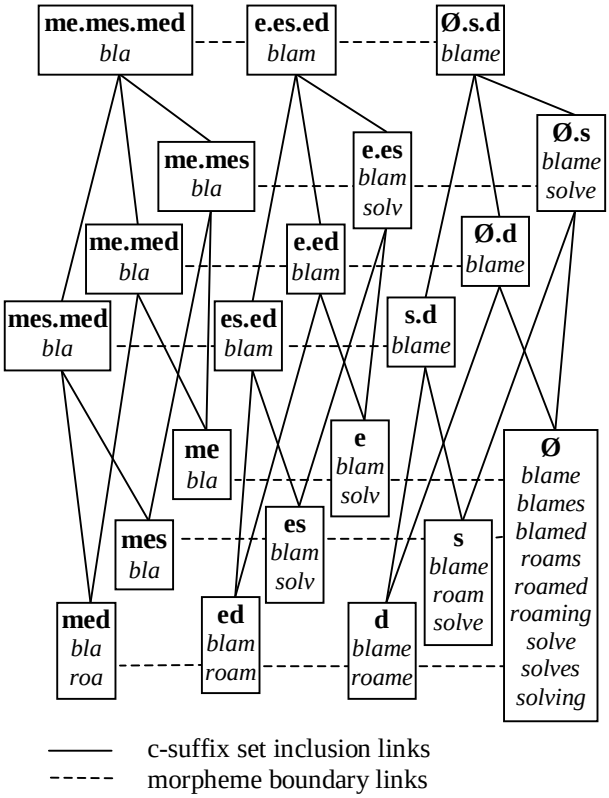

Figure 1: Portion of a CIC lattice from the toy vocabulary in Table 2

the lattice always results in a monotonic decrease in adherent size because a parent CIC requires each adherent c-stem to form a word with a superset of the c-suffixes of each child.

Horizontal morpheme boundary links, dashed lines, connect a CIC, C, with a neighbor to the right if each c-suffix in C begins with the same character. This entails that there is at most one morpheme boundary link leading to the right of each CIC. There may, however, be as many links leading to the left as there are characters in the orthography. The only CIC with depicted multiple left links in Figure 1 is **Ø**, which has left links to the CIC's **e**, **s**, and **d**. A number of left links emanating from the CIC's in Figure 1 are not shown; among others absent from the figure is the left link from the CIC **e.es** leading to the CIC **ve.ves** with the adherent *sol*. Since left links effectively divide a CIC into separate CIC's, one for each character in the orthography, adherent count monotonically decreases as left links are followed.

To better visualize what a CIC lattice looks like when derived from real data, Figure 2 contains a portion of a hierarchical lattice automatically generated from our Spanish newswire corpus. Each entry in Figure 2 contains the c-suffixes comprising the CIC, the adherent size of the CIC, and a sample of adherent c-stems. The lattice in Figure 2 covers:

1) The productive Spanish inflection class for adjectives, **a.as.o.os**, covering the four adjective paradigm cells: feminine singular, feminine plural, masculine singular, and masculine plural, respectively,
2) All possible CIC subsets of the adjective CIC, e.g. **a.as.o**, **a.os**, etc. and,
3) The imposter CIC **a.as.o.os.tro**, together with its rogue descendents, **a.tro**, and **tro**.

Other CIC's that are descendents of **a.as.o.os.tro** and that contain the c-suffix *tro* do not supply additional adherents and hence are not present either in Figure 2 or in our program's representation of the CIC lattice. The CIC's **a.as.tro** and **os.tro**, for example, both have only the one adherent, *cas*, already possessed by their common ancestor **a.as.o.os.tro**. Strictly speaking we have simplified for exposition, as the CIC **a.as.o.os.tro** is not actually present in the algorithm's representation either, because the c-stem *cas* occurred with a number of additional c-suffixes yielding the CIC:

a.as.i.o.os.sandra.tanier.ter.tro.trol.

4.2 Search

Given the framework of CIC lattices, the key task for automatic morphology induction is to autonomously separate the nonsense CIC's from the useful ones, thus identifying linguistically plausible inflection classes. This section treats the CIC lattices as a hypothesis space of valid inflection classes and searches this space for CIC's most likely to be true inflection classes in a language.

There are many possible search strategies and heuristics applicable to the CIC lattice, and while for future work we intend to explore a variety of search techniques, this paper presents a reasonable and intuitive baseline search procedure. We have investigated a series of algorithms which build upon each other. Each algorithm employs a number of parameters which are tuned by hand. These parameters are only interesting in so far as they help us find true CIC's from among the many in the lattice. The performance of each algorithm is described in section 6.

4.2.1 Vertical Only

To motivate the general approach we have taken, compare the adherent sizes of the various CIC's in Figure 2. The target CIC **a.as.o.os**, corresponding to the Spanish adjective inflection class, has 43 adherents. Its various descendents must occur with monotonically increasing adherent sizes, but frequently a child will not more than double or triple its immediate parent's adherent size, and never is there a difference greater than a factor of ten. Notice also the large adherent counts

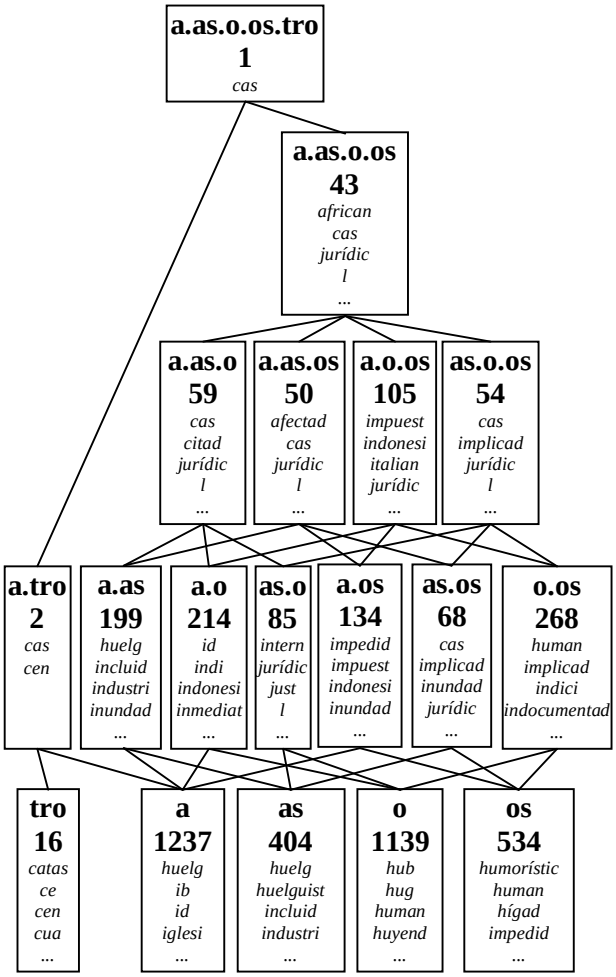

Figure 2: Hierarchical CIC lattice automatically derived from Spanish

of the level one descendents of **a.as.o.os**, the smallest is **as** with 404 adherents.

Contrast this behavior with that of CIC's involving the spurious suffix *tro*. The CIC **a.as.o.os.tro** occurs in the corpus with exactly one adherent, *cas*. Additionally, the word forms *cena*, supper, and *centro*, center, occur yielding the CIC **a.tro** with two adherents. In total *tro* is the final string of only 16 individual word forms.

In general, we expect that true suffixes in a language will both occur frequently and occur attached to a large number of stems which also accept other suffixes from the same inflection class. These considerations led us to propose three parameters for our basic search strategy:

L1 SIZE:	A level one adherent size cutoff
TOP SIZE:	An absolute adherent size cutoff
RATIO:	A parent-to-child adherent size ratio cutoff

The L1 SIZE parameter requires a c-suffix to be frequent, while the TOP SIZE and RATIO parameters require a suffix to be substitutable for other c-suffixes in a reasonable number of c-stems.

We apply these three parameters by beginning our search at the bottom of the lattice. Each level one CIC with an adherent count larger than L1 SIZE is placed in a list of path CIC's. Then for each path CIC, C, we remove C from the list of path CIC's, and in turn consider each of C's hierarchical parents, P_i. If P_i's adherent size is at least TOP SIZE, and if the ratio of P_i's adherent size to C's adherent size is larger than RATIO, then P_i is placed in the list of path CIC's. If no parent of C can be placed in the list of path CIC's, and if C's level is greater than one, then C is placed in a list of selected CIC's. When there are no more CIC's in the list of path CIC's, the search ends and the CIC's in the selected list are the CIC's the algorithm believes are true CIC's of the language.

As an illustration suppose we explored the lattice in Figure 2 with the following parameter settings:

L1 SIZE: 100
TOP SIZE: 2
RATIO: 0.1

Our search algorithm begins by comparing the adherent size of each level one CIC to L1 SIZE. The only level one CIC with an adherent count less than 100 is **tro** with 16 adherents, preventing **tro** from being placed in the list of path CIC's.

Each of the surviving level one CIC's is then considered in turn. The algorithm comes to the CIC **a**, where the ratios of adherent sizes between each of its parents **a.tro**, **a.as**, **a.o**, and **a.os** and itself are 0.002, 0.161, 0.173, and 0.108 respectively. Each of these ratios, except that between **a** and **a.tro**, at 0.002, is larger than 0.1. And since the adherent sizes of **a.as**, **a.o**, and **a.os** are each larger than TOP SIZE, these three CIC's are placed in the list of path CIC's.

From this point, every hierarchical link in Figure 2 leading to the CIC **a.as.o.os** passes the TOP SIZE and RATIO cutoffs. Thus the algorithm reaches a state where the only CIC in the list of path CIC's is **a.as.o.os**. When this good CIC is removed from the list of path CIC's, the algorithm finds that its only parent is **a.as.o.os.tro** with its lone adherent. Since TOP SIZE requires a parent to have at least two adherents, **a.as.o.os.tro** cannot be placed in the list of path CIC's. As no parent can be placed in the list of path CIC's, **a.as.o.os** is placed in the list of selected CIC's—which is the desired outcome. The list of path CIC's is now empty and the search ends.

4.2.2 Horizontal Blocking

To improve performance over the Vertical Only algorithm we next incorporated knowledge from the horizontal morpheme boundary links. Monson (2004) describes how morpheme boundary links in a CIC lattice can be thought of as branchings in a vocabulary trie where identical subtries are conflated. Harris (1955) discusses how the branching count in a suffix trie can be exploited to identify morpheme boundaries. We extend the spirit of Harris' work in our algorithm through the use of two search parameters:

HORIZ RATIO: A cutoff over:

$$\text{argmax}_c \frac{\#\text{ of adherents ending in character } c}{\text{adherent size}}$$

HORIZ SIZE: An adherent size cutoff

Left Blocking

In the first variant of horizontal blocking we apply these two horizontal parameters when considering a CIC, C, removed from the list of path CIC's. If the adherent size of C is larger than HORIZ SIZE and the maximum percentage of adherents of C that end in any one character is larger than HORIZ RATIO, then C is simply thrown out.

For example, suppose we used the following horizontal parameter settings:

HORIZ RATIO: 0.5
HORIZ SIZE: 10

The CIC **da.do** in our Spanish corpus has 62 adherents, 46, or a fraction of 0.742, of which end in the character *a* (*ada* and *ado* fill the feminine and masculine past participle cells for the *-ar* verb inflection class). If our Left Blocking search algorithm reached the CIC **da.do,** it would be discarded because while its adherent size is larger than HORIZ SIZE more than half of its adherents end with the same character. Notice that this algorithm does not explicitly follow leftward morpheme boundary links. The rationale for this behavior is that **ada.ado** will likely be explored independently by a separate vertical path. In future experiments we intend to investigate the effect of ensuring that the CIC to the left is explored by overtly placing the leftward CIC in the list of path CIC's.

Right Blocking

So far we have only described an algorithm to block paths where the correct morpheme boundary is to the left of the current hypothesis. There are also CIC's where a morpheme boundary should be moved to the right. The CIC **cada.cado** with seven adherents is one such.

Accordingly, whenever we encounter a CIC, C, all of whose c-suffixes begin with the same character (e.g. *c* in **cada.cado**) our algorithm poses the question, if we were considering the CIC to the right (e.g. **ada.ado**) would we have triggered Left Blocking? If Left Blocking would not have been

triggered then we throw C out. In other words, we prefer the rightmost possible morpheme boundary, unless there is some reason to believe the morpheme boundary should be to the left.

Taking a closer look at **cada.cado**, the CIC to its right, **ada.ado**, has 46 adherents of which the character *c* ends the most, 7 or a fraction of 0.152. If we were using a HORIZ RATIO of 0.5 as in the previous section, Left Blocking would not be triggered from **ada.ado** and so Right Blocking *is* triggered, throwing out **cada.cado**. On the other hand, if we were considering blocking **ada.ado**, where both c-suffixes begin with *a*, the HORIZ RATIO parameter would need to be larger than 0.742 before right blocking would throw out **ada.ado**.

Right Blocking Recursive

In addition to standard Right Blocking we explored recursively looking at the next most right neighbor of a CIC if the immediate right neighbor falls below the HORIZ SIZE threshold. The rationale behind this variant stems from CIC's such as **icada.icado** with 4 adherents, *crit*, *publ*, *ratif*, and *ub*. Since **icada.icado**'s immediate right neighbor **cada.cado** has only 7 adherents itself we may not want to base our blocking decision on so little data. Instead we consider the CIC **ada.ado**, discussed in the previous section, which has a large enough adherent size that we might feel confident in our judgment.

Full Horizontal Blocking

The final version of the search we tried was to combine Left Blocking and Right Blocking Recursive while constraining both to use the same values for the HORIZ RATIO and HORIZ SIZE parameters.

5 Evaluation

To evaluate the performance of the various baseline search strategies, we first decided on a standard set of six inflection classes for Spanish: two for nouns, ***Ø.s*** and ***Ø.es***, one for adjectives, ***a.as.o.os***, and three for verbs, corresponding to the traditional *-ar*, *-er*, and *-ir* verb inflection classes. We call these six inflection classes our set of standard IC's. We make no claim as to the truth or completeness of the set of standard inflection classes we used in this evaluation. The standard IC's we compiled were simply some of the most common suffixes filling some of the most common morphosyntactic properties marked in Spanish.

We then defined measures of recall, precision, and fragmentation over these standard IC's (Figure 3). As defined, recall measures the fraction of unique suffixes in the standard IC's that are found within those selected CIC's that are subsets of some inflection class in the standard; precision measures the fraction of unique suffixes among all the selected CIC's that are found within those selected CIC's that are subsets of an inflection class in the standard; and fragmentation measures redundancy, specifically calculating the ratio of the number of selected CIC's that are subsets of standard IC's to the number of inflection classes in the standard. High values for recall and precision are desirable, while a fragmentation of exactly 1 implies that the number of usefully selected CIC's is the same as the number of inflection classes in the standard.

$$\text{Recall} = \frac{\left| \bigcup_{\substack{C \in \text{ selected CIC's} \\ I \in \text{ standard IC's}}} \text{elements of } C \text{ if } C \subset I \right|}{\left| \bigcup_{I \in \text{ standard IC's}} \text{elements of } I \right|}$$

$$\text{Precision} = \frac{\left| \bigcup_{\substack{C \in \text{ selected CIC's} \\ I \in \text{ standard IC's}}} \text{elements of } C \text{ if } C \subset I \right|}{\left| \bigcup_{C \in \text{ selected CIC's}} \text{elements of } C \right|}$$

$$\text{Fragmentation} = \frac{\sum_{\substack{C \in \text{ selected CIC's} \\ I \in \text{ standard IC's}}} \begin{cases} 1 \text{ if } C \subset I \\ 0 \text{ if } C \not\subset I \end{cases}}{|\text{standard IC's}|}$$

Figure 3: Three performance measures to optimize

6 Results and Error Analysis

For each of the search variants described in section 4.2 we executed a by-hand search over the relevant parameters for those settings that optimize the F_1 measure (the harmonic mean of recall and precision). The best performing parameter settings are presented in Table 3 while quantitative results using these settings are plotted in Figure 4.

Examining the performance of each algorithm (Figure 4) reveals that the simple Vertical only search achieves a high precision at the expense of a low recall measure. The simple Vertical search also gives the smallest fragmentation, which, when combined with the high precision score, indicates a conservative algorithm that selects few CIC's. The parameter settings which achieve the highest F_1 for Left Block alone and Right Block alone each produce much higher recall than the simple Vertical search. Right Block Recursive increases precision significantly over simple Right Block and achieves

Algorithm	TOP SIZE	L1 SIZE	RATIO	HORIZ RATIO	HORIZ SIZE
V	2	192	0.3		
LB	2	64	0.1	0.3	3
RB	2	27	0.2	0.5	27
RBR	2	27	0.05	0.5	243
FHB	2	27	0.2	0.3	3

Table 3: Hand tuned optimal parameter settings for each search algorithm: **V**ertical, **L**eft **B**locking, **R**ight **B**locking, **R**ight **B**locking **R**ecursive, and **F**ull **H**orizontal **B**locking

the highest F_1 measure of any search variant. While Full Horizontal Block also performs well, sharing the values of HORIZ RATIO and HORIZ SIZE forced a compromise between Left Block and Right Block Recursive that did not significantly outperform either algorithm alone.

Of the 83 unique suffixes in the hand compiled standard inflection classes, 21 did not share a c-stem with any other c-suffix in the Spanish newswire corpus used for this evaluation—placing an upper limit on recall of 0.75 for the search algorithms presented in this paper.

Examining the parameter settings that yielded the highest F_1 measure for each search variant (Table 3) is also enlightening. Early experiments

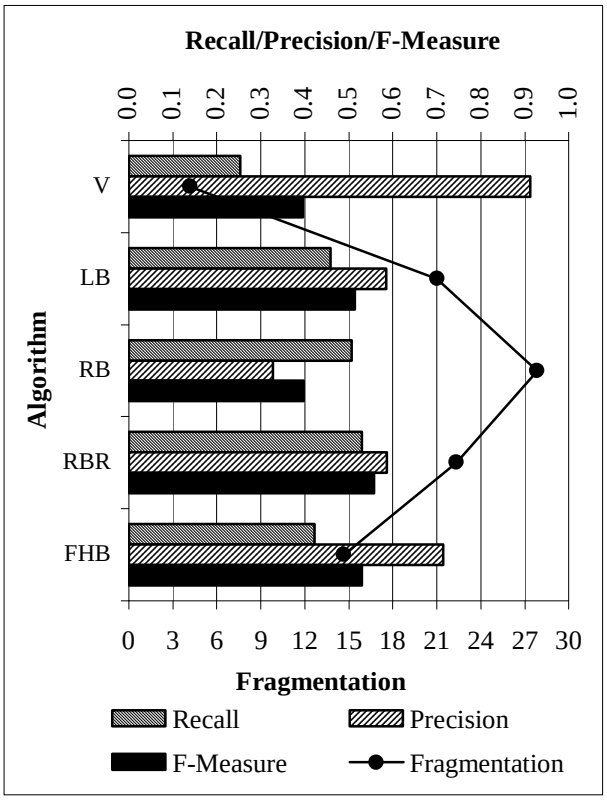

Figure 4: Recall, Precision, F_1 and Fragmentation Results for each search algorithm: **V**ertical, **L**eft **B**locking, **R**ight **B**locking, **R**ight **B**locking **R**ecursive, and **F**ull **H**orizontal **B**locking

with Vertical only search clearly demonstrated that a TOP SIZE of two, or restricting the CIC's permitted to be selected to those with at least two adherents, always resulted in better performance than other possible settings. A TOP SIZE of one places no restriction on the adherent size of a CIC, rampantly selecting CIC's, such as the level 10 CIC given at the end of section 4.1, that consist of many c-suffixes that happen to validly concatenate onto a single c-stem—obliterating reasonable precision. Higher settings for TOP SIZE induce a graceful degradation in recall. Thus all experiments reported here used a TOP SIZE of two.

Beyond TOP SIZE the only parameters available to the basic Vertical algorithm are L1 SIZE and RATIO, which provide only crude means to halt the search of bad paths. In particular, if a level one CIC, C, has more than L1 SIZE adherents, and has some parent which passes the RATIO cutoff, then some ancestor of C *will* be selected by the algorithm as a good CIC. Hence, the Vertical only algorithm ensures search gets off on the right foot by using the highest values for the L1 SIZE and RATIO parameters of any algorithm variant. Performance falls off quickly above L1 SIZE settings of 192, indicating that this parameter in this algorithm is sensitive to the size of the training corpus.

In contrast, the horizontal blocking search algorithms have additional parameters available to cull out bad search paths, and can hence afford to use lower (and more stable) values for L1 SIZE and RATIO. Recall that the Left Blocking algorithm discards paths determined to be using a morpheme boundary too far to the right, while the Right Blocking algorithm discards paths using morpheme boundaries too far to the left. Notice that since, as reasoned in section 4.1, adherent count monotonically decreases as morpheme boundary links are followed to the left, if the L1 SIZE cutoff blocks a particular CIC, C, all CIC's to the left of C will also be blocked. From these facts it follows that a large L1 SIZE will reject some paths resulting from morpheme boundaries chosen too far to the left, which would otherwise have been pursued in the Left Blocking algorithm. The Right Blocking algorithm, however, receives no such benefit, and achieves its best performance by maximizing recall with a small L1 SIZE.

Examining the best performing parameter values for the Right Blocking Recursive algorithm reveals a curious behavior in which low values for L1 SIZE and RATIO allow a permissive vertical search while stringent values of HORIZ RATIO and, particularly, HORIZ SIZE constrain the search. One explanation for these facts might be that following the monotonically increasing chain of CIC adherent sizes along right horizontal links allows the algorithm to

N	Adj	Verbs			Vertical
		ar	er	ir	23 of 23 Selected CIC's
•					Ø.s
		•			a.aba.ada.adas.ado.ar.as
		•			a.aba.ada.ado.ando.ar
		•			a.aba.ada.ado.ar.ará.en.ó
					a.*aciones.ación* .ada.adas.ar.aron
		•			a.ada.adas.ado.ar.ará
		•			a.ada.adas.ar.aron.ó
		•			a.ada.ado.ar.aron.ará.ó
		•			a.ada.ado.ar.ará.arán.en.ó
		•			a.ada.ado.ar.o.ó
		•			a.ada.ados.ar.aron.ó
		•			a.ado.ar.ara.aron.e.ó
		•			a.ado.ar.aron.ó
		•			a.an.ar.ó
	•				a.as.o.os
	•	•	•	•	a.as
		•			aba.ado.ando.ar.aron.ará
		•			aba.ado.ar.aron.ará.en
		•			ada.ado.ados.ar.aron.ó
		•			ada.ado.ando.ar.aron.ó
		•			ada.ado.ar.ará.o.ó
		•			ada.ado.ar.en.o.ó
		•			ado.ar.aron.ará.arán.en

N	Adj	Verbs			Right Blocking Recursive
		ar	er	ir	23 of 204 Selected CIC's
					Ø.ba.n.ndo
		•			a.aba.ado.ados.ar.ará.arán
					a.*aciones.ación* .adas.ado.ar
		•			a.ada.adas.ado.ar.ará
		•			a.adas.ado.an.ar
		•			a.ado.ados.ar.ó
		•			a.ado.an.arse.ó
		•			a.ado.aron.arse.ó
		•			aba.ada.ado.ar.o.os
					aciones.ación .ado.ados
					aciones .ado.ados.ará
					ación .ado.an.e
		•			ada.adas.ado.ados.aron.ó
		•			ada.ado.ados.ar.o
					ado.*adores* .o
		•			ado.ados.arse.e
		•			ado.ar.aron.arse.ará
					do.dos.ndo.r.ron
			•	•	e.ida.ido
			•		emos.ido.ía.ían
				•	ida.ido.idos.ir.ió
				•	ido.iendo.ir
				•	ido.ir.ro

Table 4: All of the CIC's selected by the conservative Vertical search algorithm (left), and a random sample of CIC's selected by the algorithm with best F_1 measure, Right Blocking Recursive (right). For each CIC row, a dot is placed in the columns representing standard IC's for which that CIC is a subset. The c-suffixes in *italics* are in no standard IC.

make intelligent blocking decisions backed by sufficient data.

The best performing parameter values for the Full Horizontal Search are a compromise between the well performing values for the Left Blocking and those for the Right Blocking algorithms. This parameter value compromise does not draw benefit from the recursion in the Right Block Recursive algorithm, but instead employs Right Block as a replacement for the relatively higher L1 SIZE parameter in the Left Blocking algorithm.

It is also interesting to examine CIC's selected by the search algorithms. Table 4 lists all of the CIC's selected by the conservative Vertical search algorithm together with a random sample of CIC's selected by Right Blocking Recursive, the algorithm which reached the highest F_1 measure of any algorithm variant.

Perhaps the most striking feature of Table 4 is the extent to which the CIC's overlap. Very few individual c-suffixes occur in only one CIC. Of all the CIC's in Table 4, only **Ø.s** and **a.as.o.os,** both among the CIC's selected by the Vertical algorithm, represent complete inflection classes in the standard IC's. The remaining CIC's are proper subsets of various verbal inflection classes. The overlapping nature of the selected CIC's suggests an additional step, which we do not investigate here, of conflating CIC's into a fewer number of meta-CIC's.

The only verbal inflection class for which subsets are able to pass the large L1 SIZE cutoff imposed by the Vertical search algorithm is *-ar*, the most frequent of the three major inflection classes in Spanish. The Right Blocking Recursive algorithm on the other hand identifies significant portions of all three verbal inflection classes.

The c-suffixes appearing in *italics* in Table 4 correspond to no suffix found in any standard IC. These alien c-suffixes fall into two categories.

1) The c-suffixes *aciones*, *ación*, and *adores* are noun forming derivational suffixes.
2) The remaining c-suffixes were formed by choosing a morpheme boundary too far to the right.

It is the second type of mistake that the Left Blocking search algorithm was specifically designed to address. Unfortunately naïvely combining the Right Blocking Recursive with the Left Blocking algorithm did not improve performance. We expect that by using separate horizontal pa-

rameters for left blocking and for right blocking we could combine these two algorithms in a less constrained fashion that would result in better overall performance.

7 Future Work

We believe the heuristic search strategy described in this paper can be significantly improved upon. We plan to investigate search strategies for both the vertical and horizontal links in our CIC lattices. We currently have plans to employ statistical independence and correlation tests to adjacent CIC's as a guide to search (Monson, 2004). Other search criteria we are considering are information gain and minimum description length measures.

There are also modifications to the search strategy that may significantly improve performance. For example, it may be advantageous to actively follow horizontal morpheme boundary links, instead of merely blocking paths, when a morpheme boundary error is discovered. The next immediate step we will take is to scale our implementation to investigate performance changes as we increase the size of our Spanish corpus.

The intention of this work is to produce a language independent morphology induction algorithm. Hence, we plan to apply this work to a variety of languages, both well studied resource-rich languages as well as low-density languages of interest to the AVENUE project.

8 Acknowledgements

The research reported in this paper was funded in part by NSF grant number IIS-0121631.

References

Jaime Carbonell, Katharina Probst, Erik Peterson, Christian Monson, Alon Lavie, Ralf Brown, and Lori Levin. 2002. Automatic Rule Learning for Resource-Limited MT. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas (AMTA-02)*.

Andrew Carstairs-McCarthy. 1998. "Paradigmatic Structure: Inflectional Paradigms and Morphological Classes." *The Handbook of Morphology*. Eds. Andrew Spencer and Arnold M. Zwicky. Blackwell Publishers Inc., Massachusetts, USA, 322-334.

Éric Gaussier. 1999. Unsupervised learning of derivational morphology from inflectional lexicons. In *Proceedings of ACL '99 Workshop: Unsupervised Learning in Natural Language Processing*.

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

Margaret A. Hafer and Stephen F. Weiss. 1974. Word segmentation by letter successor varieties. *Information Storage and Retrieval*, 10:371-385.

Zellig Harris. 1955. From phoneme to morpheme. *Language*, 31:190-222. Reprinted in Harris 1970.

Zellig Harris. 1967. Morpheme boundaries within words: Report on a computer test. *Transformation and Discourse Analysis Papers* 73, Department of Linguistics, University of Pennsylvania. Reprinted in Harris 1970.

Zellig Harris. 1970. *Papers in Structural and Transformational Linguistics*. D. Reidel, Dordrecht, Holland.

Alon Lavie, Stephan Vogel, Lori Levin, Erik Peterson, Katharina Probst, Ariadna Font Llitjós, Rachel Reynolds, Jaime Carbonell, and Richard Cohen. 2003. Experiments with a Hindi-to-English Transfer-based MT System under a Miserly Data Scenario. *ACM Transactions on Asian Language Information Processing (TALIP)*, to appear in 2(2).

Christian Monson. 2004. A Framework for Unsupervised Natural Language Morphology Induction. In *Proceedings of the Student Workshop at ACL-04*.

Katharina Probst, Lori Levin, Erik Peterson, Alon Lavie, and Jaime Carbonell. 2002. MT for Resource-Poor Languages using Elicitation-based Learning of Syntactic Transfer Rules. *Machine Translation, Special Issue on Embedded MT*, 17(4): 245-270.

Patrick Schone and Daniel Jurafsky. 2000. 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*, 67-72.

Patrick Schone and Daniel Jurafsky. 2001. Knowledge-free Induction of Inflectional Morphologies. In *Proceedings of the North American Chapter of the Association of Computational Linguistics*. 183-191.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the Human Language Technology Conference*, 161-168.