Unsupervised Word Segmentation for Bangla

Sajib Dasgupta and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083, USA
{sajib,vince}@hlt.utdallas.edu

Abstract

Unsupervised word segmentation is the task of segmenting words into prefixes, suffixes and roots without prior knowledge of language-specific morphotactics and morpho-phonological rules. This paper introduces a simple, yet highly effective algorithm for unsupervised word segmentation for Bangla, an Indo-Aryan language that is highly inflectional in nature. When evaluated on a set of 2511 human-segmented Bangla words, our algorithm achieves an F-score of 84%, substantially outperforming Linguistica, one of the most widely-used unsupervised morphological analyzers, by about 23%.

1 Introduction

Morphological segmentation is the task of segmenting a word into *morphemes* (i.e. prefixes, suffixes and roots), the smallest meaning-bearing elements of natural languages. Though very successful, *knowledge-based* approaches to word segmentation operate by relying on manually-designed heuristics, which require a lot of linguistic expertise and are also time-consuming to construct. As a result, research in morphological analysis has exhibited a shift from knowledge-based approaches to *unsupervised* approaches. Unsupervised word segmentation is typically composed of two steps: (1) a **morpheme induction** step in which morphemes are automatically induced from a vocabulary consisting of words taken from a large, unannotated corpus, and (2) a **segmentation** step in which a given word is segmented based on these automatically induced morphemes. Unsupervised word segmentation has achieved considerable success (e.g., Goldsmith (2001), Schone and Jurafsky (2001), Freitag (2005)). For instance, Schone and Jurafsky report F-scores of 88%, 92%, and 86% on English, German, and Dutch word segmentation, respectively. The recent Pascal Challenge, *Unsupervised Segmentation of Words into Morphemes*[1], has further intensified interest in this task, selecting as target languages English as well as two agglutinative languages[2] that have presented a lot of challenges to word segmentation researchers: Finnish and Turkish. Not surprisingly, the participants of the Challenge have achieved a higher accuracy on English than on the two agglutinative languages.

Our goal in this paper is to address the problem of unsupervised word segmentation for Bangla, an Indo-Aryan language spoken by more than 200 million people in Bangladesh and the Indian state of West Bengal. The problem of Bangla word segmentation is not only theoretically interesting but also of practical significance. From a research perspective, Bangla is highly inflectional, so it can be expected to pose similar challenges to researchers in word segmentation just like Turkish and Finnish. From a practical perspective, as Pushpak Bhattacharyya argued in the COLING/ACL 2006 Asian Language Processing panel discussion, the availability of an accurate word segmentation algorithm for morphologically rich languages could substantially reduce the amount of annotated data needed to construct practical natural language processing (NLP) tools such as part-of-speech (POS) taggers and noun phrase chunkers for these languages. Since the majority of Indian languages are morphologically rich and yet resource-scarce, Bhattacharyya's observation sug-

[1] See http://www.cis.hut.fi/morphochallenge2005/.
[2] Words in agglutinative languages are formed by concatenating morphemes.

gests that our progress in Bangla word segmentation can potentially accelerate the development of accurate NLP tools for analyzing Indian languages in the absence of large annotated corpora. Unfortunately, while unsupervised word segmentation has been extensively investigated for many European languages, the same is not true for Bangla and other Indian languages. To our knowledge, we are the first to tackle the task of unsupervised word segmentation for Bangla.

This paper presents an unsupervised word segmentation algorithm for Bangla that extends Keshava and Pitler's (2006) work by (1) removing spuriously induced morphemes using a threshold that depends on the length of a morpheme (Section 4), (2) identifying and removing *composite suffixes* from the induced list of morphemes (Section 5), and (3) introducing a novel use of frequency information for detecting *inappropriate morpheme attachments* (Section 6).

We train our unsupervised word segmentation algorithm on a vocabulary consisting of words (and their frequency of occurrences) collected from a large corpus of Bangla news articles. Unlike morphological analysis for many European languages, we do *not* perform the conventional pre-processing step of removing proper nouns from the vocabulary, primarily due to the lack of a publicly available proper noun identifier for Bangla. Nevertheless, our algorithm achieves very promising results: when evaluated on a set of 2511 human-segmented Bangla words, our algorithm achieves an F-score of 84%, substantially outperforming Linguistica (Goldsmith, 2001), one of the most widely-used unsupervised morphological analyzers, by about 23%

2 Related Work

There is considerable literature on the problem of unsupervised and minimally supervised word segmentation for English and other European languages. For instance, Goldsmith (2001) develops an unsupervised morphological analyzer based on Minimum Description Length, where the Expectation Maximization algorithm is used in an iterative process to segment a list of words taken from a given corpus using some predefined heuristics until the length of the morphological grammar converges to a minimum. He applies his algorithm to English and French but not to any agglutinative language. Motivated by Goldsmith, Creutz and Lagus's (2005) algorithm selects the most probable candidate segmentation that is computed via a *maximum a posteriori* formulation. DéJean (1998) develops a strategy for identifying the end of a stem by counting whether the number of characters following the stem exceeds some given threshold. By extending Dejean's idea to include transitional probabilities, Keshava and Pitler's (2006) system achieves the best result on the English dataset in the aforementioned Pascal Challenge on Unsupervised Word Segmentation.

Although all the systems described above are reasonably successful in identifying morphemes, they fail to identify inappropriate morpheme attachments, thus incorrectly segmenting a word like "ally" as "all+y". Schone and Jurafsky (2001) address this problem by introducing a method that uses the semantic relatedness between word pairs to judge whether an attachment is valid. When evaluated on an English dataset derived from the CELEX lexical database, their system achieves an F-score of 88.1%, which is the best result reported to date on this dataset. In contrast, we propose a different but novel idea of using relative frequency distribution to solve the attachment problem for Bangla. Whereas Schone and Jurafsky's method depends on complex co-occurrence statistics collected from the corpus to calculate the semantic relatedness, our system, which just uses corpus frequency, is just as effective but arguably much simpler.

Finally, although there has been a considerable amount of work on knowledge-based morphological analysis for Bangla (e.g. Chaudhuri et al. (1997), Bhattacharya et al. (2005), Dasgupta and Khan (2005), Dash et al. (2006)), none of these knowledge-based analyzers have been empirically evaluated. As a result, we cannot compare the performance of our unsupervised word segmentation algorithm with them.

3 Morpheme Induction

As mentioned before, the first step of unsupervised word segmentation aims to induce *prefixes*, *suffixes* and *roots* from a vocabulary consisting of words taken from a large, unannotated corpus. We rely on a fairly simple idea for morpheme induction. Assume that A and B are two character sequences and AB is the concatenation of A and B. If AB and A are both found in the vocabulary, then we extract B as a candidate suffix. Similarly, if AB and B are both found in the vocabulary, then we extract A as a candidate prefix.

Note, however, that this method will fail when applied to irregular words (for example,

সাংবাদকি (sAngbAdIk)[3] = সংবাদ (sngbAd) + কি (Ik)), where roots exhibit orthographic changes when they are attached to by morphemes (e.g. সংবাদ turns into সাংবাদ). Fortunately, irregular words only form a small percentage of the vocabulary; hence we hypothesize that given a large vocabulary, we can still induce a good list of morphemes. The rest of this section describes in more detail our morpheme induction process, which is composed of three steps.

3.1 Representing the Lexicon Using Tries

Following previous work (Goldsmith, 2001; Schone and Jurafsky, 2001), we represent the lexicon using the Trie data structure to enable efficient access of the vocabulary. Specifically, we (1) insert all the words in the vocabulary into a **Forward Trie**, and (2) reverse the characters of each word in the vocabulary and insert each reversed string into a **Backward Trie**. Figure 1 shows the Forward Trie created by inserting the following words: "□□□" (jAT), "□□□" (jmI), "□□□" (jAm), "□□□□" (jATI), "□□□□□□" (jATiTE), "□□□□" (jATE), "□□□□□" (jATEr), "□□□□□□" (jATEri), "□□□□□" (jATIIy), "□□□□□□□" (jATIIyTA) ,"□□□□□□" (jATVEd), "□□□□□□□" (jATVEdE). Here, the leftmost node is the root of the Trie; each edge corresponds to a character, and each black node indicates the end of a word.

3.2 Extracting a List of Candidate Affixes

Next, we extract a list of candidate affixes from the Forward Trie and the Backward Trie. Recall from the above that (1) if AB and A are found in the vocabulary, then B is a candidate suffix; and (2) if AB and B are found in the vocabulary, then A is a candidate prefix. Based on this idea, we can extract candidate suffixes from the Forward Trie and candidate prefixes from the Backward Trie, as described below.

To extract candidate **suffixes**, we search in the Forward Trie using Depth First Search (DFS) and extract any character sequence that comes in between black nodes as candidate suffixes. If there are multiple black nodes after the first black node, we include all the character sequence combination after the first black node as candidate suffixes. To exemplify, consider the black nodes at [a], [b], [c] and [d] in the Trie shown in Figure 1. At [b], the algorithm extracts "ে" (E), at [c] it extracts both "র"(r) and "রে"(Er), and at [d] it extracts "ই"(i), "রই"(ri) and "রেই" (Eri) as candidate suffixes. We employ the same procedure for extracting candidate **prefixes** from a Backward Trie.

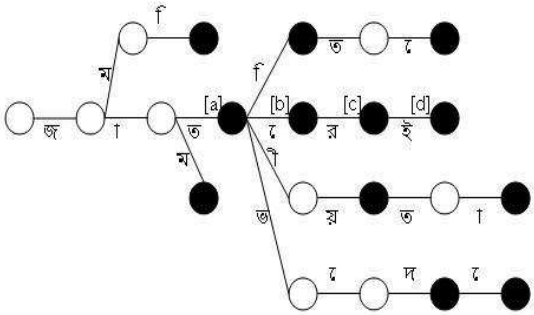

Figure 1. An Example Forward Trie

Note from the above that our procedure generates **composite** suffixes such as 'Eri'. However, it is undesirable to extract composite suffixes, because they can introduce unwanted parses (see Section 5 for details). One potential solution to this problem is to extract as candidate suffixes only those character sequences that come in between two adjacent black nodes (e.g. 'E', 'r' and 'i'), thus excluding "Er", "ri", and "Eri". However, we do want to keep suffixes (e.g. "Er") that simply subsume other suffixes (e.g. "Er" subsumes 'E' and 'r') but are not composite suffixes themselves. We will discuss in Section 5 how we detect the composite suffixes from our induced list of morphemes.

3.3 Ranking the Candidate Affixes

The above morpheme induction method can generate many spurious morphemes. To see the reason, consider the following pair of words taken from our vocabulary ["জালমে" (JAlEm), "জাল" (JAl)]. From this word pair, our algorithm would induce the candidate suffix "মে" (Em), which, however, is an erroneous suffix. To address this problem, we examine in the rest of this subsection *two* scoring metrics to score each morpheme, with the goal of assigning low scores to spurious morphemes and subsequently removing them from our list of induced morphemes.

(1) Counting the number of distinct words to which each induced morpheme attaches. In this metric, we set the score of a morpheme to be

[3] Throughout this paper we have used Romanized transliteration for Bangla which is almost phonetic. For example, 'অ' is 'a', 'আ' is '@', 'া' is 'A', 'ক' is 'k'. We have used '~' for Halant in Bangla.

the number of distinct words to which it attaches in the vocabulary. To understand the rationale behind this metric, consider the two morphemes: "রে" (Er) and "মে" (Em). "Er" attaches to 9817 distinct words in our corpus, whereas "Em" attaches to only 23. This is a good indication that "Er" is a good morpheme and "Em" is not.

(2) Incorporating the generative strength. By counting the number of distinct words to which a morpheme attaches, the first scoring metric essentially places the same weight on each word when scoring a morpheme. However, we hypothesize that some words are "better" than the others for morpheme induction, and hence a good word should be given a high weight. Specifically, we assign to each word a weight based on its **generative strength** (i.e. how many distinct induced morphemes attach to the word). In other words, if a word X attaches to 199 distinct morphemes, its strength will be 199. Given this notion of word strength, in this metric we set the score of a morpheme to be the sum of the strengths of the words to which it attaches.

To see why it makes sense to assign weights based on word strength, consider the following words in English: *scholarship*, *scholars*, *championship*, *champions*. From these words, our algorithm will infer that "hip" is a suffix. However, if we examine the words to which "hip" attaches, we can see that none of them (e.g. *scholars* and *champions*) has generative strength (i.e. they do not attach to any other suffixes). Hence, 'hip' should be given a low score, which is what we desire. As another example, consider the Bangla words: "কলজে" (klEj), "কলে" (klE), "লাগজে" (lAgEj), "লাগে" (lAgE), "আজজি" (ajIj), "আজি"(ajI), "হাউজ"(hAuj), "হাউ"(hAu). From these words, our algorithm would induce 'j' as a candidate suffix. However, since "klE", "lAgE", "ajI", and "hAu" lack generative strength, the candidate suffix 'j' should be given a low score, which is again what we desire.

Neither of the above metrics takes into account an important factor when scoring an induced affix: the *length* of the affix. As Goldsmith (2001) points out, among the induced affixes, the short ones (especially the single character affixes) are more likely to be spurious than the long ones. This is due to the fact that among different words it is easier to get one character difference at the word boundary than two or three character difference. To address this problem, Goldsmith suggests that a higher weight should be placed on longer affixes. Hence, we modify each of the scoring metrics above by multiplying the score of an affix with the length of the affix. In other words, for the first scoring metric, the score of an affix *m* is now computed as:

$score(m)=length(m)$ × (Number of different words *m* attaches to)

and for the second scoring metric, the score of an affix *m* is computed as

$$score(m)=length(m) \times \sum_{w} strength(w)$$

where *w* is a word to which *m* attaches, and $strength(w)$ is the strength of *w*.

Now, the question is: which of these two metrics should be used to score a morpheme? To address this question, we employ them separately to score the induced affixes. The two lists shown on the left half of the table are obtained by scoring each morpheme using metric 1, whereas the two list shown in the right half of the table are induced using the metric 2. Hence, by comparing the two lists, we can examine whether generative strength is indeed useful for scoring a morpheme.

As we can see from the table, after incorporating generative strength, the list does not change much for suffixes, but all the top-scoring prefixes are spurious. To investigate the reason, we examined the highest ranked prefix "পরকিল্পনা" (prIkl~pnA) and discovered that many of the words that are attached to "prIkl~pnA" are actually suffixes like "গুলো" (gUlO), "কারী" (kArII), "মতো" (mTO), "বধি" (bID), "হীণ" (hIIN). The problem here is that many suffixes in Bangla are found in the corpus as a complete meaning bearing entity, and so they work as a stem in a prefixed word. As suffixes (working like roots) generally have a high generative strength, the overall score increases manifold and so higher length prefixes (most of them are roots themselves) appear high in the rank list. In view of this problem, we employ metric 1 to score each affix, and retain an induced affix in our list if and only if its score is greater than some pre-defined threshold. Specifically, we employ a threshold of 60 and 40 for prefixes and suffixes, respectively.

3.4 Extracting a List of Candidate Roots

After filtering the spurious affixes as described in the previous subsection, we can extract an **initial** list of candidate roots using the induced list of affixes as follows. For each word, *w*, in the vocabulary, we check whether *w* can be segmented as "*r*+*s*" or "*p*+*r*", where *p* is an induced

prefix, *s* is an induced suffix, and *r* is a word in the vocabulary. If so, then *w* is *not* a root and so we do not add it to the root list; otherwise we add

Top-scoring affixes according to metric 1				**Top-scoring affixes according to metric 2**			
Prefix List		**Suffix List**		**Prefix List**		**Suffix List**	
Prefix	**Score**	**Suffix**	**Score**	**Prefix**	**Score**	**Suffix**	**Score**
bI (বি)	1054	Er (রে)	19634	prIkl~pnA (পরিকল্পনা)	23048	Er (রে)	121936
a (অ)	770	kE (কে)	13456	kOm~pAnI(কোম্পানী)	20517	kE (কে)	113584
p~rTI (প্রতী)	664	r (র)	12747	p~rTIF~xAn (প্রতিফিষন)	20240	Sh (সহ)	73184
mhA (মহা)	651	o (ও)	8213	nIr~bAcn (নির্বাচন)	20139	gUlO (গুলো)	65200
p~r (প্র)	640	I (ী)	7872	S~tEXIyAm(স্টেডিয়াম)	20016	o (ও)	56885
SU (সু)	636	Sh (সহ)	6502	p~rTIjOgITA (প্রতিযোগিতা)	19700	I (ী)	52290
@ (আ)	626	E (ে)	6218	p~rk~rIyA (প্রক্রিয়া)	19635	gUlOr (গুলোর)	52165
bIs~b (বিশ্ব)	580	dEr (দরে)	5874	SEW~cUrI(সেঞ্চুরী)	19481	E (ে)	49459
bA (বা)	544	TE (তে)	4296	anUF~xAn (অনুষ্ঠান)	18711	r (র)	48305
sIk~FA (শিক্ষা)	500	gUlO (গুলো)	3440	Sid~DAn~T (সিদ্ধান্ত)	18613	tA (টা)	44430
gN (গণ)	496	rA (রা)	3262	pAr~tnArsIp (পার্টনারশিপ)	18080	tI (টী)	44208
prI (পরী)	486	tA (টা)	2592	SmS~jA (সমস্যা)	17700	dEr (দরে)	43626

Table 1. Top N induced morphemes ranked according to the proposed scoring metrics

w to the root list. However, since Bangla words can contain multiple roots, it is possible that after stripping off the induced affixes from a word, we will end up with a string that is a concatenation of several roots. Hence, we make another pass over our initial list of roots to remove those strings that contain multiple roots.

So far we have described our basic morpheme induction algorithm. In Sections 4-6, we will propose three extensions to this basic algorithm.

4 Length-Dependent Threshold

As mentioned above, we retain an induced morpheme in our list if and only if its score is greater than some threshold. However, instead of having the same threshold for all induced morphemes, we employ a varying threshold that depends on the length of a morpheme. In particular, we apply larger thresholds for shorter morphemes. The rationale is simple: since shorter morphemes are more likely to be erroneous than their longer counterparts, it makes more sense to employ larger thresholds to shorter morphemes. We set our length-dependent threshold as follows:

Threshold of affix A = *m* * *Uniform Threshold*
Where *Uniform Threshold* is set to 40 for suffixes and 60 for prefixes, and

m = (4-length of A) if length < 4
= 1 if length >= 4

We will empirically investigate in Section 7 whether employing this varying threshold would yield better segmentation performance than employing a uniform threshold.

5 Composite Suffix Detection

A *composite suffix* is a suffix formed by combining multiple suffixes. For instance, "তাকে"(TAkE)

is a composite suffix that comprises "তা" (TA) and "কে" (kE). As mentioned above, we have to detect and remove composite affixes from the induced morpheme list because their presence can produce incorrect segmentation of words. For example, if "TAkE" is present in the suffix list then "ভদ্রতাকে" (vd~rTAkE) will be erroneously segmented as "vd~r+TAkE" (note: the correct segmentation is "vd~r+TA+kE").

Now the question is: how to detect a composite suffix? Simple concatenation of multiple suffixes does not always produce a composite suffix. For example, "Er", "E" and "r" all are valid suffixes but "Er" is not a composite suffix. Hence, we need a more sophisticated method for detecting composite suffixes. Specifically, our method posits a suffix as a composite suffix if both of the following criteria are satisfied.

Affix strength. This criterion is based on the observation that, given a composite suffix *a* formed by combining two suffixes *a*1 and *a*2, the strength of *a* (i.e. the number of different words to which *a* attaches) should be smaller than the strength of *a*1 and the strength of *a*2. To see the reason, consider the composite suffix "ments" ("ment" + "s") in English. The number of words to which "ment" or "s" attaches is far greater

than the number of words to which "ments" attaches. As another example, consider the Bangla suffix "এর" (Er). As shown in Table 2, "Er" attaches to 9817 distinct words, whereas its component suffix "E" only attaches to 6218 words in the corpus. Hence, employing affix strength enables us to correctly determine that "Er" is not a composite suffix.

Word-level similarity. This criterion is based on the observation that, if a word is attached to a composite suffix (AB), then it is highly likely that it will also be attached to first component suffix A. In other words, AB and A should be similar in terms of words to which they are attached. For example, if an English word (say "sing") is attached to "ers", it should also be attached to "er". This property does not hold for non-composite suffixes, however. For instance, words that are attached to "ent" (say "absorb") are **not** attached to "en". Given this observation, we can detect composite suffixes by first computing the similarity in between a suffix (AB) and its first component suffix (A) as follows:

$$P(A \mid AB) = \frac{|W'|}{|W|}$$

where $|W'|$ is the number of words that attach to both suffix AB and suffix A, and $|W|$ is the number of words that attach to suffix AB.

If the above probability is greater than some threshold (normally set as 0.6) and the first criterion (i.e. affix strength) is satisfied, then we posit AB as a composite suffix. One advantage of the above probabilistic metric is that it helps select the best one among multiple segmentation options. For example, "Eri" is a composite suffix that can be segmented two ways: "E+ri" and "Er+i". The similarity between "Eri" and "Er" is 0.979, which is greater than the similarity between "Eri" and "E" (0.739). So, "Er+i" is selected as a correct segmentation of composite suffix "Eri".

Most importantly, composite suffix detection has enabled correct parsing of many Bangla verbs, which arguably have a very complex morphological structure. For example, the actual segmentation of the verb "হাটছিলাম" (hAtCIlAm) is "hAt+CI+l+Am", where "hAt" is the root, "CI" is the tense (Continuous) marker, "l" is the time (Past) marker, and "Am" is the person (first person) marker. Encouragingly, our algorithm produces exactly the same output. The segmentation of "hAtCIlAm" is shown below step by step:

hAtCIlAm = hAt + CIlAm
= hAt + CI + lAm
[detection of composite suffix CIlAm]
= hAt + CI + l + Am
[detection of composite suffix lAm]

Nevertheless, the algorithm fails to parse verbs with perfect tense and future time marker. For example, the word "হাটবে" (hAtbE) is incorrectly parsed as "hAt+bE" although the correct parse should be "hAt+b+E" ('b' is a future marker in Bangla).

Suffixes determined to be composite			**Suffixes determined to be non-composite**		
Suffix	**Division**	**Word level Similarity**	**Suffix**	**Division**	**Word level Similarity**
AkE (220)	A (1764) + kE (6728)	0.954	AT (83)	A (1764) + T (340)	0.45
AnO (98)	A (1764) + nO (160)	0.70	Ar (854)	A (1764) + r (12747)	0.57
ErtA (16)	Er (9817) + tA (1296)	0.9375	IyE (116)	I (1246) + yE (325)	0.53
ITE (214)	I (1246) + TE (2148)	0.915	TA (463)	T (340) + A (1764)	0.038
TAo (82)	TA (463) + o (8213)	0.94	TE (2148)	T (340) + E (6218)	0.057
T~bEr (45)	T~b (62) + Er (9817)	0.91	Tm (85)	T (1246) + m (236)	0.023
dEri (107)	dEr (1958) + i (7872)	0.95	Tr (54)	T (346) + r (12747)	0.07
IIsh (58)	II (684) + sh (3251)	0.98	Er (9817)	E (6218) + r (12747)	0.43
krNE (27)	krN (84) + E (6218)	0.77	kE (6728)	k (332) + E (6218)	0.015
CEn (259)	CE (335) + n (1478)	0.83	nA (188)	n (1478) + A (1764)	0.4
ECI (34)	E (6218) + CI (144)	0.97	bA (64)	b (156) + A (1764)	0.2
bEn (94)	bE (147) + n (1478)	0.82	bE (55)	b (156) + E (6218)	0.47
lAm (120)	l (616) + Am (235)	0.85	bI (81)	b (156) + I (1246)	0.45
lEn (233)	l (616) + En (597)	0.86	c~CIl (22)	c~CI (20) + l (616)	0.45

Table 2. Examples of suffixes checked for compositeness. Individual suffix strengths are parenthesized.

The reason why the algorithm fails to detect "bE" as a composite suffix is that there are not enough words in the vocabulary that are attached to suffix 'b' (first person future indefinite tense form of a verb), and so the similarity value in between "bE" and "b" is low. It is worth noticing

that our Bangla corpus is composed of news articles, which are normally written in "Third Person" form. (Consider the following two sentences extracted from the corpus: (1) "প্রধান মন্ত্রী আজ জাতরি উদ্দশ্যে ভাষণ দবিনে" ("pradhan mantri aaj jatir urdershe vhashan diben"), and (2) "সংোয়বে আজ দুর্দান্ত বলংি করছে", (shoyeb aaj durdanta baling kreche) (s) where "দবিনে" (diben) and "করছে" (kreche) are the third person honorific and non-honorific representation of the corresponding root verb). Unless we have a text collection with different verb forms (first, second and third person variations), it would be very difficult to segment Bangla verbs correctly.

6 Incorrect Attachment Detection

After inducing the prefixes, suffixes and roots, the next challenge is to identify inappropriate affix attachment. Consider the English word "candidate". The affix induction and parse strategy described thus far incorrectly segment it into "candid" and "ate" (where "candid" is a stem found in the corpus and "ate" is a valid suffix).

Consequently, we propose a simple yet novel idea of using relative corpus frequency to decide whether morpheme attachment to a particular root word is plausible or not. Our idea is based on the following hypothesis: if a word, A, is a morphological inflection or derivation of a word, B, then the corpus frequency of A is likely to be less than that of B. In other words, we hypothesize that the inflectional or derivational form of a root word occurs less frequently in the corpus than the root word itself.

To obtain empirical support for our hypothesis, we show in Table 3 some **word-root frequency ratios (WRFRs)**, each of which is obtained by dividing the frequency of a Bangla word by the frequency of its root. As we can see, the corpus frequency (1670) of "নারী" (nArII) is far bigger than that of the constituent stem "nAr" (3). Hence, our hypothesis correctly predicts that the suffix "ী" (II) cannot attach to "nAr" to form "nArII". Note that, WRFR is less than 1 for all the words in the left side of the table, whereas it's greater than 1 for all the words in the right side of Table 3.

The question, then, is: to what extent does our hypothesis hold true? To investigate this question, we randomly selected 400 words from our vocabulary and removed from them the following words: (1) words that do not have morphological segmentation (e.g. "মানুষ" (mAnUsh)); (2) proper nouns (e.g. "করমি" (krIm)); (3) words whose constituent root word is absent in the vocabulary (e.g. "আসব", @sb= "@s+b" but "@s" is not found in the vocabulary); and (4) compound words (i.e. words with multiple roots). The final list contains 287 words. We then hand-segmented each of these words into Prefix+Root or Root+Suffix, and found that the WRFR is less than 1 in 83.56% of the cases (see Table 4). This provides reasonably strong evidence for our hypothesis that during attachment, the frequency of a word is less than that of its constituent root word. Among the remaining 16.44% of the words that violate our hypothesis, we found that many of them that should be segmented as "Root+Suffix" are verbal inflections. In Bangla, inflected forms of the verb roots occur more often in the corpus than the roots (e.g. "করে" (kre) occurs more often than "kr"). This can be attributed to the grammatical rule that says that the main verb of a sentence has to be inflected according to the subject in order to maintain sentence order.

Correct Attachments			**Incorrect Attachments**		
Word	**Root**	**WRFR**	**Word**	**Root**	**WRFR**
@SrEr (□□□□□)	@Sr	34/200 = 0.17	nArII (□□□□)	nAr	1670/3 = 556
@bEgE (□□□□□)	@bEg	28/71 = 0.39	JAbTIy (□□□□□□)	JAbT	198/3 = 66
jIIbnKE (□□□□□□)	jIIbn	63/908 = 0.0693	KOlA (□□□□)	KOl	587/4 = 146.75
apb~jy (□□□□□□)	b~jy	8/940 = 0.0085	jAmAyAT (□□□□□□□)	jA-mAy	996/5 = 199.2
upjATI (□□□□□□)	jATI	17/509 = 0.033	bAjAr (□□□□□)	bAj	1093/3 = 364.3
p~rTIdIn (প্র□□□□□)	dIn	728/6932 = 0.105	jbAb (□□□□)	jbA	813/3 = 271

Table 3. Word/root frequency ratios

	Root+Suffix	**Prefix+Root**	**Overall**
Number of Words	245	41	286
WRFR less than 1	84.5%	78.05%	83.56%

Table 4. Hypothesis validation

Now, we can make use of the above hypothesis, incorporating relative frequency information to improve word segmentation as follows: if a word is segmented as "Prefix+Root" or "Root+Suffix", we then check whether the corre-

sponding WRFR is greater than some predefined threshold (>1). If so, we consider the attachment erroneous and treat the whole word as a root. The threshold is set differently for prefixes and suffixes. Specifically, we set the threshold to be 2-4 for prefix attachment and 10-15 for suffix attachment [4]. The threshold for suffixes is set higher than the prefixes to account for the inflectional words (mainly verbs) which normally come more than their corresponding root forms.

7 Word Segmentation

In Sections 3-6, we described how we induce a good list of morphemes. Once we induce the morphemes, we can apply them to segment a word in the test set.

The segmentation process is fairly simple. Given a word in the test set, we identify all possible segmentations of the word using only the induced affixes and roots. Then, we filter those candidate segmentations that violate any of the simple linguistic constraints below:

- There has to be at least one root in the segmentation.
- If a morpheme is a prefix, then the immediately following morpheme should be either a root or a prefix.
- If a morpheme is a suffix, then the immediately preceding morpheme should be either a root or a suffix.

If more than one candidate segmentation remains after applying the above constraints, we take the one that has minimum number of morphemes to be the final segmentation of the word. For example, if "বালকগুলো" (bAlkgUlO) has two candidate segmentations: "bAlk+gUlO" and "bAl+k+gUl+O", then our algorithm selects the first one to be the segmentation of the word.

8 Evaluation

Now, let us evaluate our segmentation algorithm.

8.1 Experimental Setup

Vocabulary creation. The corpus from which we extract our vocabulary contains one year of news articles taken from the Bangla newspaper "Prothom Alo". Specifically, we only use articles that are sports news or editorials, as well as those that appear in the first page or the last page of the newspaper. We then pre-process each of these articles by tokenizing it and removing punctuations and other unwanted character sequences (such as "***"). The remaining words are then used to create our vocabulary, which consists of 142955 distinct words. Unlike morphological analysis for many European languages, however, we do not take the conventional step of removing proper nouns from our vocabulary, because we do not have a name entity identifier for Bangla.

Test set preparation. To create our test set, we randomly choose 3000 words from our vocabulary that are at least 3-character long. We impose this length restriction when selecting our test cases simply because words of length one or two do not have any morphological segmentation in Bangla. We then manually remove the proper nouns and words with spelling mistakes from the test set before giving it to two of our linguists for hand-segmentation. In the absence of a complete knowledge-based morphological parsing tool and a hand-tagged morphological database for Bangla, our linguists had to depend on the Bangla dictionary[5] for annotating our test cases.

There is one caveat in our manual annotation procedure, however. Many Bangla words are morphologically derived from Sanskrit roots. These words are very difficult, if not impossible, for any morphological analyzer to segment correctly, because the orthographic changes that take place during the segmentation process are highly non-linear and complex in nature. One example of such word is "□□□□□□□" (bIrUd~D), whose actual segmentation is "□□+□□□+□□□(□)" (bI+rUd+k~T (T)) – which is tough to obtain. As a result, we instruct our linguists to simplify the segmentation of these words so that the orthographic changes are within tractable edit distance. Given this restriction, the Bangla word shown above (i.e. "□□□□□□□") will simply be segmented as "□□+□□□□□" (bI+rUd~D). However, if the meaning of a segmented word differs from that of the original word, then we simply treat the original word as a root (i.e. the word should not be segmented at all). Words that fall within this category include "□□□□□□", "□□□□□", and "□□□□□□□□□" etc. After all the words have been manually segmented, we remove those for which the two linguists produce inconsistent segmentations. The resulting test set contains 2511 words.

[4] We found that the result does not change much when we vary the threshold within these ranges.

[5] The dictionaries we used are "বঙ্গীয় শব্দকোষ" (Bangiya Sabdakosh) by হরচিরণ বন্দ্যোপাধ্যায় (Haricharan Bandopaday) and "বাংলা একাডেমী ব্যবহারিক বাংলা অভিধান" (Bangla Academy Bebharic Bangla Avidan).

Evaluation metrics. We use two standard metrics --- *exact accuracy* and *F-score* --- to evaluate the performance of our morphological analyzer on the test set. Exact accuracy is the percentage of the words whose proposed segmentation is identical to the correct segmentation. F-score is simply the harmonic mean of recall and precision, as computed using the formulas below.

Precision = (H) / (H+I)

Recall = (H) / (H+D)

F-score = (2H) / (2H+I+D)

where H represents the number of morpheme boundaries correctly identified, and I and D represent the total number of Insertions and Deletions that needs to be applied to the correct output to make it identical to the proposed output. For instance, comparing the incorrect segmentation "un+fri+endly" against the correct segmentation "un+friend+ly" results in 1 Hit, 1 Insertion and 1 Deletion.

System Variations	Exact Accuracy	Precision	Recall	F-score
Baseline (Linguistica)	37.08	58.25	65.15	61.48
Basic Induction	46.67	76.66	66.2	71.04
Composite Suffix Detection	55.99	79.07	80.61	79.83
Length dependent thresholds	58.38	81.97	79.75	80.85
Incorrect attachment detection	**65.83**	89.1	80.22	**84.43**

Table 5. Results

8.2 Results

The baseline system. Following previous work (Schone and Jurafsky, 2001), we use Goldsmith's (2001) Linguistica[6] as our baseline system for unsupervised morphological learning. The first row of Table 5 shows the results of our baseline system on the test set when it is trained on the Bangla corpus described in Section 8.1 (with all the training parameters set to their default values). As we can see from Table 5, the exact accuracy is about 37%, which is poor to say the least. We presume the poor result is due to the inability of Linguistica to handle Bangla compound words and its complex verbal inflectional system. Nevertheless, the baseline achieves a decent F-measure of 61.48%.

[6] Linguistica is publicly available at http://humanities.uchicago.edu/faculty/goldsmith/Linguistica2000/

Our segmentation algorithm. Results of our segmentation algorithm are shown in rows 2-5 of Table 6. Specifically, row 2 shows the results of our basic segmentation algorithm. Rows 3-5 show the results when composite suffix detection (see Section 5), length-dependent thresholds (see Section 4), and incorrect attachment detection (see Section 6) are added to the basic system one after the other. It is worth mentioning that (1) our basic algorithm already outperforms the baseline system by a wide margin in terms of both exact accuracy and F-score; and (2) while each of our additions to the basic algorithm boosts system performance, composite suffix detection and incorrect attachment detection contribute to performance improvements particularly significantly. As we can see, the best segmentation performance is achieved when all of our three additions are applied to the basic algorithm. We also perform 5-fold cross-validation on our test set and found that the F-scores at each level are statistically significant at p=0.05.

8.3 Discussion and Error Analysis

As part of the analysis of our word segmentation algorithm, we are interested in examining whether it can correctly segment complicated test cases. Encouragingly, our system successfully segments complex verbal inflections like "□□□□□□□□□□" (dUlIyECIl) as "dUl+IyE+CI+l", as well as multi-root words like "বনি·াদনকন্দ্রগুল·াও" (bInOdnkEndRgUlOo), whose correct segmentation is "bInOd+n+kEndR+gUlO+o". Even more interestingly, it correctly parses English words, which are widely used in the Sports section of the newspaper. For example, words like "□□□□" (blIng) and "□□□□□□□□□□"(FAinAlIS~t) are correctly segmented into "bl+Ing" and "FAinAl+IS~t". It is worth mentioning that the compounding nature of Bangla and the influence of foreign languages have introduced into our repository a lot of new words, whose presence increases the difficulty of the segmentation task. Nevertheless, our word segmentation system manages to stem those words correctly.

We also investigated the words that were incorrectly segmented by our system. The errors can be broadly divided into following categories:

(1) Verbal inflections. These constitute a large portion of the words incorrectly segmented by our algorithm. There are two reasons for such errors. First, the root of an incorrectly segmented verb is missing from the corpus. (Hence, "উঠা"

(uthA) is incorrectly segmented because its root "উঠ" (uth) is not found in the corpus, for instance.) Second, the first and second person forms of verbs are often missing in the corpus, as the newspaper articles from which our vocabulary is induced contain mostly third person forms of verbs.

(2) Irregular words. When root words exhibit orthographic spelling changes during attachment, our system fails to identify the roots. For example, "রক্ি্সারহী" is not correctly segmented, because the root "আরহী" (@rhII) is changed into "ারহী" (ArhII) during attachment.

(3) Incorrect attachments. Although we use relative frequency to detect incorrect morpheme attachments, many incorrect prefixations and suffixations remain undetected (e.g. "শকিল" (sIkl) is a root word but it is incorrectly parsed as "sIk+l"). This suggests that we need a more sophisticated system for incorrect morpheme attachment detection.

(4) Unseen roots. Many words remain unsegmented because their constituent root words are absent in the corpus. For example, the root "নতে্" (nETR) in "নতে্ত্ব" (nETRT~b) is not found in our corpus.

9 Conclusions and Future Work

We have presented a new unsupervised algorithm for Bangla word segmentation that, when evaluated on a set of 2511 human-segmented Bangla words, substantially outperforms Goldsmith's Linguistica. Analysis reveals that our novel use of relative frequency information, together with our proposed technique for composite suffix detection, have contributed to the superior performance of our algorithm.

In future work, we plan to investigate whether our algorithm can be improved by incorporating automatic irregular word form detection (cf. Yarowsky and Wicentowski (2000)) and using automatically acquired information about the semantic relatedness between word pairs (cf. Schone and Jurafsky (2001)). In addition, we plan to build a POS tagger for Bangla that exploits the morphological information provided by our algorithm. This contrasts with existing work on POS tagging for Indian languages, where POS taggers are commonly built by using information provided by knowledge-based word segmentation algorithms (e.g. Singh et al. (2006)).

Acknowledgements

We would like to thank the three anonymous reviewers for their valuable comments. We would also like to give special thanks to Dr. Mumit Khan, Mr. Kamrul Haider and Mr. Naushad Uzzaman of Centre for Research on Bangla Language Processing (CRBLP), BRAC University, Bangladesh for allowing us to use the "Prothom Alo" Corpus and other linguistic resources.

References

Samit Bhattacharya, Monojit Choudhury, Sudeshna Sarkar, and Anupam Basu. 2005. Inflectional Morphology Synthesis for Bangla Noun, Pronoun and Verb Systems. In *Proc. of the National Conference on Computer Processing of Bangla (NCCPB 05)*, pages 34 - 43.

Michael R. Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. In *Machine Learning, 34*, pages 71-106.

Bidyut Baran Chaudhuri, Niladri Sekhar Dash, and P. K. Kundu. 1997. Computer Parsing of Bangla Verbs. In *Linguistics Today,* Vol.1, No.1, pages 64-86.

Mathias Creutz and Krista Lagus. 2005. Unsupervised Morpheme Segmentation and Morphology Induction from Text Corpora Using Morfessor 1.0. Publications in *Computer and Information Science, Report A81, Helsinki University of Technology.*

Niladri Sekhar Dash. 2006. The Morphodynamics of Bengali Compounds decomposing them for lexical processing. In *Language in India (www.languageageinindia.com)*, Vol 6:7.

Sajib Dasgupta and Mumit Khan. 2004. Feature Unification for Morphological Parsing in Bangla. In *Proceeding .of International Conference on Computer and Information Technology.*

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

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

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

S. Keshava and E. Pitler. 2006. A simpler, intuitive approach to morpheme induction. In Proc. of the *PASCAL Challenge Workshop on Unsupervised Segmentation of Words into Morphemes.*

Kimmo Koskenniemi. 1983. Two-level morphology: a general computational model for word-form recognition and production. *Publication No. 11. Hel-*

sinki: University of Helsinki Department of General Linguistics.

Patrick Schone and Daniel Jurafsky. 2001. Knowledge-free induction of inflectional morphologies. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics* (NAACL).

Smriti Singh, Kuhoo Gupta, Manish Shrivastava, and Pushpak Bhattacharyya. 2006. Morphological Richness Offsets Resource Demand – Experiences in Constructing a POS Tagger for Hindi. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics* (ACL).

David Yarowsky and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics* (ACL), pages 207-216.