Lang Resources & Evaluation
DOI 10.1007/s10579-007-9031-y

Unsupervised morphological parsing of Bengali

Sajib Dasgupta · Vincent Ng

Abstract Unsupervised morphological analysis is the task of segmenting words into prefixes, suffixes and stems without prior knowledge of language-specific morphotactics and morpho-phonological rules. This paper introduces a simple, yet highly effective algorithm for unsupervised morphological learning for Bengali, an Indo–Aryan language that is highly inflectional in nature. When evaluated on a set of 4,110 human-segmented Bengali words, our algorithm achieves an F-score of 83%, substantially outperforming Linguistica, one of the most widely-used unsupervised morphological parsers, by about 23%.

Keywords Morphological parsing · Word segmentation · Data annotation · Unsupervised learning · Asian language processing · Bengali

1 Introduction

While research in Asian language processing has gained a lot of momentum in the past decade, much of this research effort has indeed been focusing on only a handful of *oriental* languages such as Chinese, Korean, and Japanese. On the other hand, being spoken by more than 200 million people residing mostly in Bangladesh and the Indian state of West Bengal, Bengali is far less computerized than any of these oriental languages. However, with the rapid increase in the amount of Bengali data

S. Dasgupta (✉) · V. Ng
Human Language Technology Research Institute, University of Texas at Dallas,
Richardson, TX 75083, USA
e-mail: sajib@hlt.utdallas.edu

V. Ng
e-mail: vince@hlt.utdallas.edu

available in electronic form, there is a practical need for developing automatic tools for processing Bengali.

Bengali, a member of the Indo–Aryan language family, has several linguistic characteristics that can potentially complicate its automatic processing. First, the Bengali morphology is very productive, especially for verbs, with each root verb taking more than 50 different forms. In addition, the Bengali lexicon contains a large number of compound words, i.e., words that have more than one root, which can be created from almost any combination of nouns, pronouns and adjectives. The large vocabulary as a result of its morphological richness makes it difficult to manually construct a Bengali lexicon. Second, Bengali is more or less free word order (even though subject–object–verb is the typical word order), thus making its syntactic analysis potentially more difficult than that for fixed order languages such as English. Finally, the fact that all Bengali letters have only one case complicates the detection of proper nouns in Bengali than in languages with both upper and lower case letters.

This paper addresses a fundamental problem in Bengali language processing: *morphological parsing* (also known as *word segmentation*). The goal of morphological parsing is to segment a given word into the smallest meaning-bearing elements known as *morphemes*. For instance, the English word "unforgettable" can be divided into three morphemes: "un", "forget", and "able", whereas the Bengali word "অনাধুনকিতার" (anAdhUnIkTAr)[1] can be divided into "an" (Prefix), "AdhUnIk" (Root), "TA" (Suffix), and "r" (Inflection). While computational morphology has been extensively studied for many European languages, this has not been the case for Bengali.

Our goal in this paper is to investigate an *unsupervised* approach to Bengali morphological parsing, which, to our knowledge, represents the first attempt at applying unsupervised learning to this Bengali language processing problem. Unsupervised morphological parsing is typically composed of two steps: (1) a **morpheme induction** step in which morphemes are first automatically acquired 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 induced morphemes. The biggest challenge in unsupervised morphological parsing, then, lies in the ability to induce morphemes correctly without prior knowledge of language-specific morphotactics and morpho-phonological rules. It is worth noticing, though, that unsupervised morphological parsing has achieved considerable success for many European languages (e.g., Goldsmith 2001; Schone and Jurafsky 2001; Creutz 2003; Freitag 2005; Cavar et al 2006). For instance, Schone and Jurafsky report F-scores of 88%, 92%, and 86% on English, German, and Dutch word segmentation, respectively. Nevertheless, empirical evaluations in the recent PASCAL Challenge, *Unsupervised Segmentation of Words into Morphemes*,[2] reveal that the success of unsupervised word segmentation algorithms does not carry over to agglutinative languages such as

[1] Throughout this paper, we use the Romanized transliteration for Bengali, which is almost phonetic. For example, 'অ' is 'a', 'আ' is '@', 'া' is 'A', 'ক' is 'k', 'ট' is 't', 'ত' is 'T', 'ঠ' is 'th', etc. We have used '~' for Halant in Bengali. Our transliteration mapping table is shown in our data distribution site at http://www.utdallas.edu/~sajib/dataset.html

[2] See http://www.cis.hut.fi/morphochallenge2005/

Finnish and Turkish,[3] both of which have presented significant challenges to word segmentation researchers because of their morphological richness. Being highly inflectional in nature, Bengali is expected to offer similar challenges to researchers as Finnish and Turkish.

Not only is Bengali morphological parsing a challenging research problem, its solution is of practical significance. As Pushpak Bhattacharyya argues 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 language processing tools such as part-of-speech taggers for these languages. Since Bengali, like the majority of Indo–Aryan languages, is morphologically rich and yet resource-scarce, Bhattacharyya's observation suggests that our progress in Bengali morphological parsing can potentially accelerate the development of automatic tools for analyzing Bengali and other Indo–Aryan languages in the absence of large annotated corpora.

The major contribution of this paper is the introduction of a morphological parser for Bengali. Specifically, our parser extends Keshava and Pitler's (2006) algorithm,[4] the best performer for English in the aforementioned PASCAL Challenge, with three new techniques (see Sects. 4–6) that focus on improving the segmentation of regular words.[5] The key features of our algorithm are:

The algorithm is totally unsupervised: As mentioned above, there have been very few attempts at tackling the Bengali morphological parsing problem (e.g., Chaudhuri et al. 1997; Bhattacharya et al. 2005; Dasgupta and Khan 2004; Dash 2006), all of which have adopted *knowledge-based* approaches. These approaches operate by segmenting a word using manually-designed heuristics, which require a lot of linguistic expertise and are also time-consuming to construct. Worse still, these heuristics are typically language-specific, implying that a new set of heuristics has to be designed for each new language encountered. On the other hand, our algorithm is unsupervised, relying solely on *language-independent* techniques for morpheme induction. To our knowledge, we are the first to apply unsupervised learning to morphological parsing of an Indo–Aryan language.

The algorithm can segment words with multiple roots: Many existing segmentation algorithms can only be applied to words with one root and one suffix (e.g., DéJean 1998; Snover and Brent 2001). Goldsmith (2001) relaxes this severe limitation by allowing words with multiple affixes to be segmented correctly. Creutz (2003) moves one step further by enabling the segmentation of words with multiple roots, thus facilitating morphological parsing of agglutinative languages. Our algorithm, like Creutz's, is capable of segmenting words with multiple prefixes, suffixes and roots, as a Bengali word can be composed of a lengthy sequence of alternating roots and affixes.

[3] A word in an agglutinative language is composed of a linear sequence of distinct morphemes.

[4] Keshava and Pilter's algorithm has been applied to English, Finnish, and Turkish only.

[5] Our morphological parser does not handle the segmentation of words that show orthographic character changes during attachment with other morphemes. Nevertheless, since less than 4% of our test cases correspond to words in this category, not handling them will unlikely lead to a dramatic degradation of system performance.

The algorithm identifies inappropriate morpheme attachments: Many existing morphological parsers erroneously segment "ally" as "all + y", because they fail to identify that the morpheme "y" should *not* attach to the word "all". Schone and Jurafsky (2001) represents one of the very few attempts at addressing this *inappropriate morpheme attachment* problem. Specifically, they introduce a method that exploits the semantic relatedness between word pairs to judge whether the attachment of a morpheme to a root is valid, and show that identifying inappropriate attachments can substantially improve performance. On the other hand, we propose in this paper a novel use of relative frequency distribution to solve the attachment problem. Whereas Schone and Jurafsky's method relies on complex co-occurrence statistics for calculating semantic relatedness, our system, which just uses word frequency, is shown to be effective in improving segmentation performance and is arguably much simpler.

When evaluated on a set of 4,110 hand-segmented Bengali words chosen randomly from a news corpus, our segmentation algorithm achieves an F-score of 83%, substantially outperforming Linguistica (Goldsmith 2001), one of the most widely-used unsupervised morphological parsers, by about 23% in F-score. Unlike ours, none of the existing Bengali morphological parsers has been evaluated empirically, presumably due to the lack of annotated datasets. In fact, the lack of annotated datasets has been a major obstacle to the computerization of resource-scarce languages such as Bengali. Hence, we believe that our dataset would be a valuable addition to the list of resources publicly available for Bengali language processing,[6] facilitating comparative evaluation of different Bengali word segmentation algorithms.

The rest of this paper is organized as follows. Section 2 presents related work on unsupervised morphological parsing. In Sect. 3, we describe our basic algorithm for inducing morphemes from our Bengali vocabulary. Sections 4–6 present three extensions to this basic morpheme induction algorithm. In Sect. 7, we describe our algorithm for segmenting a word in the test set using the automatically acquired morphemes. We then evaluate the efficacy of our approach in Sect. 8 and conclude with future work in Sect. 9.

2 Related work

As mentioned in the introduction, the problem of unsupervised and minimally supervised morphological learning has been extensively studied for English and many other European languages. In this section, we will give an overview of the three major approaches to this problem.

One common approach to unsupervised morphological learning is to first identify morpheme boundaries and then identify the morphemes. For instance, Harris (1955) develops a strategy for identifying morpheme boundaries that checks whether the number of different letters following a sequence of letters exceeds some given threshold. Hafer and Weiss (1974) improve Harris's algorithm by proposing 15 different heuristics that depend on successor and predecessor frequencies to identify

[6] Our dataset is available at http://www.utdallas.edu/∼sajib/dataset.html

morpheme boundaries. Their best heuristic achieves a precision of 0.91 and recall of 0.61 on an English corpus of approximately 6,200 word types, which is very small compared to the number of word types typically seen in existing literature on unsupervised morphological induction. DéJean (1998) improves Harris's segmentation algorithm by first inducing a list of the 100 most frequent morphemes and then using those morphemes for word segmentation. The aforementioned PASCAL Challenge on Unsupervised Word Segmentation undoubtedly intensified interest in this problem. Among the participating groups, Keshava and Pitler's (2006) segmentation algorithm combines the ideas of DéJean and Harris and achieves the best result on the English dataset.

Another approach to unsupervised morphological learning is based on an application of the Minimum Description Length (MDL) principle. The goal is to find a set of morphemes such that when each word in a given corpus is segmented according to these morphemes, the total length of an encoding of the corpus is minimized. Specifically, the Expectation Maximization (EM) algorithm is used to iteratively 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. Brent et al. (1995) introduce an information-theoretic notion of compression to represent the MDL framework, although the overall aim of their work is to find an appropriate set of suffixes from a corpus rather than the correct morphological analysis of each word. They use the n most common words in the Wall Street Journal corpus of the Penn Treebank to induce the suffix list, where n ranges from 500 to 8,000. Brent (1999) and Snover and Brent (2001) later propose a Bayesian Model for MDL that yields very few false suffixes over a wide range of input sizes in English and French. Goldsmith (1997) tries to find the segmentation point of a word based on the probability and length of the hypothesized stems and affixes. In a subsequent paper, Goldsmith (2001) adopts the MDL approach and provides a new information-theoretic compression system that gives a measure of the length of the morphological grammar. He applies his algorithm to English and French and reports accuracies of 82.9% and 83.3% respectively. He also groups together the possible suffixes for each stem, and introduces the signature paradigm that is helpful for determining syntactic word classes (i.e., part-of-speech classes). Motivated by Goldsmith, Creutz (2003) and Creutz and Lagus (2005) propose a probabilistic maximum *a posteriori* formulation that uses prior distributions of morpheme length and frequency to measure the goodness of an induced morpheme. They work on English and Finnish (a highly agglutinative language) and report better accuracy than Goldsmith's Linguistica morphological parser.

The last approach, introduced by Freitag (2005), first automatically clusters the words using local co-occurrence information and then induces the suffixes according to the orthographic dissimilarity between the words in different clusters. His segmentation algorithm achieves a high precision (0.95) when morphemes are induced from an English vocabulary that consists of the 10 K most frequent terms in the Wall Street Journal corpus of the Penn Treebank. He also makes the interesting observation that employing a larger vocabulary size (say 20 K) for morpheme induction considerably degrades system precision and recall (0.8 and 0.82, respectively).

3 The basic morpheme induction algorithm

As mentioned in the introduction, our unsupervised morphological parser is composed of two steps: (1) inducing *prefixes*, *suffixes* and *roots* from a vocabulary consisting of words taken from a large, unannotated corpus, and (2) segmenting a word based on these induced morphemes. The biggest challenge in unsupervised morphological learning lies in accurately performing step 1 (i.e., morpheme induction). This section describes our morpheme induction method.

3.1 Extracting a list of candidate affixes

The first step of our morpheme induction method involves extracting a list of candidate affixes. We rely on a fairly simple idea originally proposed by Keshava and Pitler (2006) for extracting candidate prefixes and suffixes. 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. Following previous work (e.g., Goldsmith 2001; Schone and Jurafsky 2001), we represent the vocabulary using the Trie data structure to allow efficient extraction of affixes.

3.2 Ranking the candidate affixes

The above affix induction method is arguably overly simplistic, and therefore can generate many spurious affixes. To exemplify, consider the English word pair: "diverge" and "diver". From this word pair, our algorithm would induce the candidate suffix "ge", which, however, is erroneous. The same problem occurs for Bengali. For example, our algorithm would induce from the word pair ["জালমে" (JAlEm), "জাল" (JAl)] the candidate suffix "মে" (Em), which again is an erroneous suffix. To address this problem, we examine in the rest of this subsection *two* scoring metrics to score each affix, with the goal of assigning low scores to spurious affixes and subsequently removing them from our list of induced affixes.

Metric 1: Counting the number of word types to which each induced affix attaches. In this metric, we set the score of an affix to be the number of word types to which it attaches in the vocabulary. To understand the rationale behind this metric, consider the two suffixes in Bengali: "রে" (Er) and "মে" (Em). "Er" attaches to 9817 word types in our corpus, whereas "Em" attaches to only 23. This indicates that "Er" is a good affix and "Em" is not.

Metric 2: Incorporating the generative strength. By counting the number of word types to which an affix attaches, metric 1 essentially places the same weight on each word when scoring an affix. However, some words are "better" than the others for morpheme induction (e.g., words to which many different affixes attach), 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 affixes

attach to the word). Given this notion of word strength, in metric 2 we set the score of an affix 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 basic morpheme induction algorithm will infer that "hip" is a suffix. However, if we examine the words to which "hip" attaches (e.g., "scholars" and "champions"), we can see that none of them has generative strength (i.e., no other suffixes attach to these words). Hence, this scoring metric will assign a low score to "hip", which is what we desire. As another example, consider the Bengali words: "কলজে" (klEj), "কলে" (klE), "লাগজে" (lAgEj), "লাগে" (lAgE), "আজজি" (ajIj), "আজি"(ajI), "হাউজ"(hAuj), and "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 scoring metric will assign a lower score to the candidate suffix "j", 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

$$\text{score}(m) = \text{length}(m) \times (\text{Number of different words to which } m \text{ attaches})$$

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

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

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

To investigate which of these two scoring metrics is better, we employ them separately to score the induced affixes. The top-scoring prefixes and suffixes according to metric 1 are shown on the left half of Table 1. All the affixes in both the prefix list and the suffix list are correct, and in fact they represent the most commonly used affixes in Bengali.

Next, we examine the top-scoring prefixes and suffixes according to metric 2 (shown in the right half of Table 1). After incorporating generative strength, we can see that the suffix list does not change much, but surprisingly, all the top-scoring prefixes are spurious. A closer examination of the affix lists also reveals that metric 1 is better scoring metric than metric 2: 78% of the top 50 prefixes induced by metric 1 are correct, whereas the corresponding accuracy for metric 2 is only 11%. To investigate the reason, we examined the highest ranking prefix "পরকিল্পনা" (prIkl∼pnA) and discovered that many of the words to which "prIkl∼pnA" attaches are actually suffixes like "গুলো" (gUlO), "কারী" (kArII), "মতো" (mTO), "বধি" (bID) and "হীণ" (hIIN).

Table 1 Top N-scoring affixes according to metric 1 (left) and metric 2 (right)

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	1,054	Er	19,634	prIkl∼pnA	23,048	Er	121,936
a	770	kE	13,456	kOm∼pAnI	20,517	kE	113,584
p∼rTI	664	r	12,747	p∼rTIsh∼thAn	20,240	Sh	73,184
mhA	651	O	8,213	nIr∼bAcn	20,139	gUlO	65,200
p∼r	640	I	7,872	S∼tEdhIyAm	20,016	o	56,885
SU	636	Sh	6,502	p∼rTIjOgITA	19,700	I	52,290
@	626	E	6,218	p∼rk∼rIyA	19,635	gUlOr	52,165
bIs∼b	580	dEr	5,874	SEn∼cUrI	19,481	E	49,459
bA	544	TE	4,296	anUsh∼thAn	18,711	r	48,305
sIk∼shA	500	gUlO	3,440	Sid∼Dan∼T	18,613	tA	44,430
gN	496	rA	3,262	pAr∼tnArsIp	18,080	tI	44,208
prI	486	tA	2,592	SmS∼jA	17,700	dEr	43,626

The problem here is that many suffixes in Bengali are found in the corpus as a complete meaning bearing entity, and so they work as a stem in a prefixed word. As a suffix (working like a stem) generally has a high generative strength, the overall score increases manifold and longer prefixes appear high in the list.

Hence, we conclude that metric 1 does a better job at scoring candidate affixes than metric 2. Hence, in our basic morpheme induction algorithm, we will 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. These thresholds are determined based on a small validation set consisting of 500 hand-segmented Bengali words that are randomly chosen from our corpus.[7]

3.3 Extracting a list of candidate roots

After filtering the spurious affixes as described in the previous subsection, we 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 w to the root list. However, since Bengali 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

[7] We expect that larger thresholds are needed for languages that have a larger vocabulary (e.g., Turkish and Finnish) because an affix is likely to be generated from a larger number of words.

another pass over our initial list of roots to remove those strings that contain multiple roots.

3.4 Extensions to the basic induction algorithm

So far, we have described our basic morpheme induction algorithm. For each of the following three sections, we will propose an extension to this basic induction algorithm. Specifically, in Sect. 4, we will discuss an extension that involves *employing a length-dependent threshold*. Sections 5 and 6 present our second extension (i.e., *detecting composite suffixes*) and our third extension (i.e., *improving root induction*), respectively.

4 Employing a length-dependent threshold

Let us begin by motivating our first extension, *length-dependent threshold*. Recall from Sect. 3.2 that, in our basic morpheme induction algorithm, 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 will employ a varying threshold that depends on the length of a morpheme. In particular, we use larger thresholds for shorter morphemes. The rationale is simple: since shorter morphemes (especially those that are of length 1 and 2) are more likely to be erroneous than their longer counterparts, it makes more sense to employ larger thresholds for shorter morphemes. We set our length-dependent threshold as follows:

$$\text{Threshold for affix A} = m \times C,$$

where C is a constant set to 40 for suffixes and 60 for prefixes as in Sect. 3.2

$$\begin{aligned} \text{and } m &= (4 - \text{ length(A)}) \text{ if length(A)} \leq 2 \\ &= 1 \text{ if length(A)} > 2 \end{aligned}$$

We will empirically investigate in Sect. 8 whether employing this varying threshold would yield better segmentation performance than employing a length-independent threshold.

5 Detecting composite suffixes

Our second extension to the basic morpheme induction algorithm involves the detection of **composite suffixes**. 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) (like "ers" in English which is formed by "er" and "s"). However, not all suffixes formed by combining multiple suffixes are composite. For instance, "রে" (Er) is a **non-composite** suffix in Bengali, even though it comprises the two simple suffixes "ে"(E) and "র"(r).

Our goal is to detect and remove composite suffixes from the list of morphemes induced using our basic algorithm, because their presence can produce incorrect segmentation of words. For example, if the composite suffix "TAkE" is present in the induced morpheme list, then "ভদ্রতাকে" (vd∼rTAkE) will be erroneously segmented as "vd∼r + TAkE" (note: the correct segmentation is "vd∼r + TA + kE"). The reason is that the presence of the composite suffix causes the segmentation algorithm to believe that "TAkE" is a non-divisible unit, leading to under-segmentation.

Now the question is: How to detect a composite suffix? Not all strings that can be segmented into two suffixes are actually composite suffixes. As we have seen at the beginning of this section, "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.

Suffix strength: This criterion is motivated by the observation that, given a composite suffix a formed by combining two suffixes $a1$ and $a2$, the strength of a (i.e., the number of different words to which a attaches) should be smaller than the minimum of the strength of $a1$ and the strength of $a2$. As an example, consider the composite suffix "fullness" ("full" + "ness") in English. The number of words to which "full" or "ness" attaches is far greater than the number of words to which "fullness" attaches in a naturally-occurring corpus. Consider the *non-composite* Bengali suffix "Er". It attaches to 9,817 word types in our corpus, but its component suffix "E" only attaches to 6,218 words. Hence, this suffix violates the suffix strength criterion and is correctly predicted to be non-composite. However, there are suffixes like "AT" and "Ar" (see the right column of Table 2) that satisfy the suffix strength criterion and yet are not composite. This illustrates why using suffix strength alone is not sufficient for determining the compositeness of a suffix.

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

$$\text{Similarity}(AB, A) = P(A|AB) = \frac{|W'|}{|W|}$$

where |W′| is the number of words to which both AB and A attach, and |W| is the number of words to which AB attaches.

In other words, the similarity between the two suffixes, AB and A, is the probability of seeing A conditioned on seeing AB. If this probability is greater than some threshold (we set it to 0.6) and the first criterion (i.e., suffix strength) is

Table 2 Examples of suffixes checked for compositeness

Suffixes determined to be composite			Suffixes determined to be non-composite		
Suffix	Division	Similarity	Suffix	Division	Similarity
AkE (220)	A (1,764) + kE (6,728)	0.954	AT (83)	A (1,764) + T (340)	0.45
AnO (98)	A (1,764) + nO (160)	0.70	Ar (854)	A (1,764) + r (12,747)	0.57
Ei (1,274)	E (6,218) + i (7,872)	0.96	IyE (116)	I (1,246) + yE (325)	0.53
Eri (445)	Er (9,817) + i (7,872)	0.979	TA (463)	T (340) + A (1,764)	0.038
Tao (82)	TA (463) + o (8213)	0.94	TE (2,148)	T (340) + E (6,218)	0.057
T∼bEr (45)	T∼b (62) + Er (9817)	0.91	Tm (85)	T (1,246) + m (236)	0.023
dEri (107)	dEr (1,958) + i (7,872)	0.95	Tr (54)	T (346) + r (12,747)	0.07
krNE (27)	krN (84) + E (6218)	0.77	kE (6,728)	k (332) + E (6,218)	0.015
CEn (259)	CE (335) + n (1,478)	0.83	nA (188)	n (1,478) + A (1,764)	0.4
ECI (34)	E (6,218) + CI (144)	0.97	Er (9,817)	E (6,218) + r (12747)	0.43
bEn (94)	bE (147) + n (1,478)	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**

The strength of each suffix is parenthesized

Composite suffixes that are incorrectly identified as non-composite are boldfaced

satisfied, then we posit AB as a composite suffix. One advantage of the above probabilistic metric is that it can potentially be used to select the best segmentation of a word among multiple candidates. For example, "রেই" (Eri) is a composite suffix that can be segmented as either "E + ri" (the incorrect segmentation) or "Er + i" (the correct segmentation). Since the similarity between "Eri" and "Er" (0.979) is greater than that between "Eri" and "E" (0.739), "Er + i" is more likely to be the correct segmentation of "Eri".

Most importantly, composite suffix detection has enabled us to segment many Bengali verbs with complex morphology correctly. 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. Below we show how our algorithm segments "hAtCIlAm" step by step:

$$\begin{aligned} \text{hAtCIlAm} &= \text{hAt} + \text{CIlAm} \\ &= \text{hAt} + \text{CI} + \text{lAm} \quad \text{[detection of composite suffix CIlAm]} \\ &= \text{hAt} + \text{CI} + \text{l} + \text{Am} \quad \text{[detection of composite suffix lAm]} \end{aligned}$$

.

To investigate how reliable suffix strength and word-level similarity are with respect to detecting composite suffixes, we (1) apply these two criteria to all the suffixes that are concatenations of multiple suffixes, and (2) determine which are composite suffixes and which are not. Results for a randomly selected set of suffixes are shown in Table 2, where the left column lists the suffixes identified by our

criteria as composite, and the right column lists the suffixes that are identified as non-composite.

Note that all the entries in the left column are indeed valid composite suffixes in Bengali. In addition, all but the last three entries ("bE", "bI" and "c∼CIl", which are different tense markers in Bengali) in the right column are valid non-composite suffixes. Failure to detect these three and similar tense markers has resulted in incorrect segmentations of present or past continuous and future indefinite forms of Bengali verbs. For example, the word "হাটবে" ("hAtbE", future tense, third person form of verb "hAt") is under-segmented as "hAt + bE" (note: the correct segmentation is "hAt + b + E"). The reason why the algorithm fails to detect "bE" as a composite suffix is that there are not enough words in the vocabulary to which the suffix "b" (first person, future indefinite tense form of a verb) attaches, and so the similarity value between "bE" and "b" is low (0.47).

The question, then, is: Why are there not enough words in the vocabulary to which the suffix "b" attaches? The reason can be attributed to the fact that "b" is a first-person marker, but the Bengali corpus from which we extracted our vocabulary is composed of news articles, which are normally written in "Third Person" form. Unless we have a text collection with different verb forms (first, second and third person variations), it would be very difficult to segment Bengali verbs correctly.

6 Improving root induction

Our third extension to the basic morpheme induction algorithm involves improving the root induction method described in Sect. 3.3. One potential problem with this root induction method is *low recall*: many words in the vocabulary that are roots are not present in our induced root list. To see the reason, consider again the induction method applied to the English word "candidate". Assuming, without loss of generality, that "candidate" and "candid" are found in the vocabulary and "ate" is an induced suffix, the root induction method will incorrectly segment "candidate" as "candid + ate"; as a result, it does not consider "candidate" as a root. So, to improve the root induction method, we should prevent the segmentation of words like "candidate". One way to do this is to determine that the attachment of the suffix "ate" to the root "candid" to form "candidate" is incorrect.

Now, the question is: How can we determine whether morpheme attachment (e.g., "ate") relative to a particular root word (e.g., "candid") is correct or not? In this section, 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 (i.e., A is formed by attaching an affix *m* to B), then the 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.[8]

[8] Note that in many inflectional languages, the root form rarely stands alone, and so the morphologically formed A is likely to be more frequent than its root form. However, from a computational perspective, it is beneficial to exploit this hypothesis in our segmentation algorithm, as it applies to a fairly large percentage of words.

Table 3 Some word-root frequency ratios (WRFRs)

Examples of correct attachments			Examples of incorrect attachments		
Word	Root	WRFR	Word	Root	WRFR
@SrEr (আসরের)	@Sr	34/200 = 0.17	nArII (নারী)	nAr	1,670/3 = 556
@bEgE (আবগে)	@bEg	28/71 = 0.39	JAbTIy (যাবতীয়)	JAbT	198/3 = 66
jIIbnKE (জীবনকে)	jIIbn	63/908 = 0.0693	KOlA (খোলা)	KOl	587/4 = 146.75
Apb~jy (অপব্যয়)	b~jy	8/940 = 0.0085	jAmAyAT (জামায়াত)	jAmAy	996/5 = 199.2
upjATi (উপজাতি)	jATi	17/509 = 0.033	bAjAr (বাজার)	bAj	1,093/3 = 364.3
p~rTIdIn (প্রতিদিন)	dIn	728/6,932 = 0.105	jbAb (জবাব)	jbA	813/3 = 271

To obtain empirical support for our hypothesis, we show in Table 3 some randomly chosen Bengali words with their **word-root frequency ratios (WRFR)**, each of which is obtained by dividing the frequency of a word by the frequency of its root. The word-root pairs in the left side of the table are examples of correct attachments, whereas those in the right side are not. Consider the word "নারী" (nArII) in the right side of the table; the WRFR of "nArII" and "nAr" is 556, which means the corpus frequency of "nArII" (1670) 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 is 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 selected 400 words from our vocabulary that can be segmented as Prefix + Root or Root + Suffix and removed (1) proper nouns and (2) words whose constituent root word is absent in the vocabulary thus lacking root frequency information (e.g., "আসব", @sb= "@s + b" but "@s" is not found in the vocabulary). The final list contains 287 words. We then hand-segmented each of these words into Prefix + Root or Root + Suffix, and computed the WRFR ratio for each word-root pair. We found that the WRFR is less than one in 83.56% of the 257 words. 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 Bengali, 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.

Since we have shown that our hypothesis is correct to a fairly large extent, we can now use relative frequency information to identify incorrect morpheme attachments and improve root induction. Specifically, we incorporate relative frequency information in our basic root induction method as follows: For each word, w, in our vocabulary, we check (1) whether w can be segmented into any of $r + s$ or $p + s$ pattern, where p and s are valid prefixes and suffixes respectively and r is another word in the vocabulary, and (2) whether WRFR in between w and r is less than some predefined threshold (>1).

If w satisfies both constraints, it means that w is segmentizable, and hence we **do not add** w to the list of induced roots. Otherwise, we add w into the list of roots. The WFFR threshold is set differently for prefixes and suffixes. Specifically, we set the threshold to be 2 for prefix attachment and 10 for suffix attachment. (Note, however, that the result is not sensitive to small changes to these thresholds.) We employ a higher threshold for suffixes than prefixes to account for the fact that inflectional words (mainly verbal suffixations) normally occur more frequently than their corresponding root forms.

7 Word segmentation

In Sect. 3–6, we described how we induce a good list of affixes and roots. After inducing the morphemes, we can use them to segment a word in the test set into a sequence of morphemes, $m_1\ m_2\ ...\ m_n$, by adopting a *generate-and-remove* strategy, as described below.

Given a word w in the test set, we (1) **generate** all possible segmentations of w using only the induced affixes and roots, and then (2) apply a sequence of tests to **remove** candidate segmentations until we are left with only one candidate, which we take to be the final segmentation of w.

Our first test involves removing any candidate segmentation $m_1 m_2\ ...\ m_n$ that violates any of the linguistic constraints below:

(1) At least one of $m_1, m_2, ..., m_n$ is a root.
(2) For $1 \leq i < n$, if m_i is a prefix, then m_{i+1} must be a root or a prefix.
(3) For $1 < i \leq n$, if m_i is a suffix, then the m_{i-1} must be a root or a suffix.
(4) m_1 can not be a suffix and m_n can not be a prefix.

Next, we apply our second test, in which we retain only those candidate segmentations that have the smallest number of morphemes. For example, if "বালকগুলো" (bAlkgUlO) has two candidate segmentations: "bAlk + gUlO" and "bAl + k + gUlO", then we select the first one to be the segmentation of w.

If more than one candidate segmentation still remains, we apply our third test to remove any candidate c that *satisfies* one of the three cases below.

Case 1: There exists a root r in c such that r is immediately preceded by a prefix p and immediately followed by a suffix s, but neither the substring pr nor the substring rs is in our vocabulary.
Case 2: There exists a root r in c such that r is immediately preceded by a prefix p but *not* immediately followed by a suffix, and the substring pr is *not* in our vocabulary.
Case 3: There exists a root r in c such that r is immediately followed by a suffix s but not immediately preceded by a prefix, and the substring rs is *not* in our vocabulary.

As an example of applying the third test described above, consider segmenting the Bengali word "আরবতিে" (@rbITE). This word has two candidate segmentations ("@rb + I + TE" and "@rb + IT + E"), both of which follow the Root + Suffix + Suffix pattern. Since "@rbI" is in our vocabulary whereas "@rbIT" is not, we remove

"@rb + IT + E" from our list of candidate segmentations (because the second case is satisfied) but retain "@rb + I + TE" (because none of the three cases is satisfied).

If more than one candidate still remains, we score each remaining candidate using the heuristic below, selecting the highest-scoring candidate to be the final segmentation of *w*. Basically, we score each candidate segmentation by summing up the *strength* of each morpheme in the segmentation, where (1) the strength of a prefix/suffix is simply the number of word types in the vocabulary to which the prefix/suffix attaches, multiplied by the length of the prefix/suffix, and (2) the strength of a root is the number of distinct morphemes that attach to it, again multiplied by the length of the root. For example, the word "আচরণে" (@crNE) has two segmentation options: "@crN + E" and "@c + rNE". The strengths of the morphemes "@crN", "E", "@c" and "rNE" are 80, 5937, 26 and 33, respectively. So we select "@crN + E" as the final segmentation, because it has the highest strength (6,017=80 + 5,937).

8 Evaluation

In this section, we evaluate our morphological parsing algorithm.

8.1 Experimental setup

Vocabulary creation: The corpus from which we extract our vocabulary contains one year of news articles taken from the Bengali newspaper *Prothom Alo*. Specifically, we only use articles that are sports news or editorials, as well as those that appear in the first page and the last page of the newspaper.[9] 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 1,42,955 word types. 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 Bengali.

Test set preparation: To create our test set, we randomly choose 5,000 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 Bengali. 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 Bengali, our linguists had to depend on two Bengali dictionaries[10] for annotating our test cases.

[9] These are the major sections of Prothom Alo. The remaining sections are relatively small and are simply ignored.

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

There is one caveat in our manual annotation procedure, however. Many Bengali words are morphologically derived from Sanskrit roots.[11] 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 "বিরূদ্ধ" (bIrUd∼D), whose actual segmentation is "বি+রূধ+ক্ত(ত)" (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 Bengali word shown above (i.e., "বিরূদ্ধ") will simply be segmented as "বি+রূদ্ধ" (bI + rUd∼D). However, if the meaning derived from the 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 "প্রধান" (p∼rdhAn), "আবেদন" (@bEdn), and "প্রতিবেদন" (p∼rTIbEdn), for instance. After all the words have been manually segmented, we remove those for which the two linguists produce inconsistent segmentations. The resulting test set contains 4,110 words.

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

$$\text{Precision} = (\text{H})/(\text{H} + \text{I})$$
$$\text{Recall} = (\text{H})/(\text{H} + \text{D})$$
$$\text{F-score} = (2\text{H})/(2\text{H} + \text{I} + \text{D})$$

where H is the number of Hits (i.e., correctly placed boundaries), and I, D represent the number of morpheme boundaries needed to be inserted into and deleted from S_c, respectively, to make it identical to S_p. For instance, comparing the incorrect segmentation "un + fri + endly" against the correct segmentation "un + friend + ly", we obtain 1 Hit, 1 Insertion and 1 Deletion, thus yielding a F-score of 0.5 and an exact accuracy of 0. Note that most previous work simply reports results in terms of F-score, which is a less stringent evaluation metric than exact accuracy. However, we believe that reporting results in terms of both metrics will give us a better picture of the strengths and weaknesses of a morphological parser.

8.2 Results

The baseline system: Following Schone and Jurafsky (2001), we use Goldsmith's (2001) Linguistica[12] as our baseline system for unsupervised morphological learning. The first row of Table 4 shows the results of our baseline system on the test set when it is trained on the Bengali corpus described in Sect. 8.1 (with all the training parameters

[11] Sanskrit roots have compact orthography which is not morpho-phonologically transparent. That is, one written unit does not necessarily correspond to one morpheme or syllable.

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

Table 4 Results. The best exact accuracy and F-score are highlighted.

System variations	Exact accuracy (%)	Precision (%)	Recall (%)	F-score (%)
Baseline	36.32	58.23	63.27	60.63
Basic induction	47.05	76.14	65.15	70.22
Length dependent thresholds	48.95	78.37	65.47	71.34
Detecting composite suffixes	58.66	79.44	82.1	80.75
Improving root induction	**64.62**	86.64	80.02	**83.19**

set to their default values). As we can see, the exact accuracy is about 36%. On the other hand, the baseline achieves a decent F-score of 60.63%. This indicates that many of the analyses returned by Linguistica are only partially correct rather than exactly correct. A closer examination of Linguistica's output reveals that it is particularly weak at segmenting Bengali compound words and its complex verbal inflectional system.

Our segmentation algorithm: Results of our segmentation algorithm are shown in rows 2–5 of Table 4. Specifically, row 2 shows the results of our segmentation algorithm when used in conjunction with the basic morpheme induction methods described in Sects. 3.1–3.3. Rows 3–5 show the results when our techniques for employing length-dependent thresholds, detecting composite suffixes, and improving root induction are incorporated into 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 evaluation metrics; and (2) while each of our additions to the basic algorithm boosts system performance, composite suffix detection and improved root induction 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.[13] We also performed 5-fold cross validation and found that each addition to the system improves performance statistically significantly at $p = 0.05$.

8.3 Discussion and error analysis

As part of the analysis of our algorithm, we examine whether our morphological analyzer can handle complicated test cases. We found that our system successfully segments complex verbal inflections like "দুলয়িছেলি" (dUlIyECIl) as "dUl + IyE + CI + l", and multi-root words like "বণিঃোদনকন্েদ্রগুলঃোও" (bInOdnkEndRgUlOo) as "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 "বলিং" (blIng) and "ফাইনালস্িট"(FAinAlIS∼t) are correctly segmented as "bl + Ing" and "FAinAl + IS∼t", respectively. It is worth mentioning that

[13] It may seem that our performance improvements over Linguistica have come from our fine-tuning the thresholds. However, our system has achieved good performance on English, Turkish and Finnish using almost the same set of thresholds. The only exception is the thresholds used for inducing affixes (see Sect. 3); however, these thresholds can be set automatically depending on the vocabulary size of a language (see Dasgupta and Ng (2007)).

the compounding nature of Bengali 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 morphological parser manages to stem those words correctly.

We also examined 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. For instance, "উঠা" (uthA) is incorrectly segmented because its root "উঠ" (uth) is not found in the corpus. 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, "রিক্সারহী" (rIk∼sArhII) 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 algorithm 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 "নতৃে" (nETR) in "নতৃেত্ব" (nETRT∼b) is not found in our corpus.

9 Conclusions and future work

We have presented a new unsupervised algorithm for Bengali morphological parsing. Our work distinguishes itself from previous algorithms for Bengali morphological parsing in two important aspects. First, all previous algorithms adopt knowledge-based approaches, thus requiring a lot of time and linguistic expertise to implement. Second, none of them has been empirically evaluated, and hence it is unclear how well they perform. Despite its simplicity, our algorithm achieves very promising results: when evaluated on a set of 4,110 human-segmented Bengali words, the algorithm achieves an F-score of 83% and an exact accuracy of 66%, outperforming Goldsmith's Linguistica by 23% in F-score and 28% in exact accuracy. Analysis reveals that our novel use of relative frequency information, together with our technique for composite suffix detection, have contributed to the superior performance of our algorithm.

In future work, we intend to improve our algorithm in a number of ways. First, we will examine the problem of morphologically analyzing highly irregular word forms. This involves automatically acquiring transformation rules that specify what characters are inserted or deleted during the transformation, and is considered a challenging problem even for morphologically impoverished languages such as English (Yarowsky and Wicentowski 2000). Second, we plan to employ automatically

acquired information about the semantic relatedness between word pairs (see Schone and Jurafsky 2001) to improve our incorrect attachment detection algorithm. Finally, motivated by Singh et al.'s (2006) work on Hindi, we plan to investigate how to build a part-of-speech tagger for Bengali that exploits the morphological information provided by our algorithm.

Bengali language processing is still in its infancy. As mentioned in the introduction, one major obstacle to the computerization of Bengali is the scarcity of annotated corpora. As part of our commitment to developing high-performance tools and algorithms for automatically analyzing Bengali, we intend to construct annotated datasets for different Bengali language processing problems. With annotated data, we hope to advance the state of the art in Bengali language processing by (1) enabling empirical evaluations of Bengali language processing systems, and (2) tackling problems in Bengali language processing using corpus-based techniques, which are by far the most successful techniques in natural language learning. Above all, we hope to stimulate interest in the computerization of Bengali in the natural language processing community.

References

Bhattacharya, S., Choudhury, M., Sarkar, S., & Basu, A. (2005). Inflectional morphology synthesis for Bengali noun, pronoun and verb systems. In *Proceedings of the national conference on computer processing of Bangla (NCCPB 05)*, pp. 34–43.

Brent, M. R. (1999). An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning, 34*, 71–106.

Brent, M. R., Murthy, S. K., & Lundberg, A. (1995). Discovering morphemic suffixes: A case study in minimum description length induction. In *Proceedings of the fifth international workshop on artificial intelligence and statistics*.

Cavar, D., Rodriguez, P., & Schrementi, G. (2006). Unsupervised morphology induction for part-of-speech-tagging. In *Penn working papers in Linguistics: Proceedings of the 29th annual Penn Linguistics colloquium*, Vol. 12.1.

Chaudhuri, B. B., Dash, N. S., & Kundu, P. K. (1997). Computer parsing of Bangla verbs. In *Linguistics Today, 1*(1), 64–86.

Creutz, M. (2003). Unsupervised segmentation of words using prior distributions of morph length and frequency. In *Proceedings of the 41st annual meeting of the ACL*, pp. 280–287.

Creutz, M., & Lagus, K. (2005). Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0. In Computer and information science, Report A81, Helsinki University of Technology.

Dasgupta, S., & Khan, M. (2004). Feature unification for morphological parsing in Bangla. In *Proceedings of international conference on computer and information technology*.

Dasgupta, S., & Ng, V. (2007). High-performance, language-independent morphological segmentation. In NAACL-HLT 2007: *Proceedings of the main conference*, pp. 155–163.

Dash, N. S. (2006). The Morphodynamics of Bengali Compounds decomposing them for lexical processing. *Language in India (www.languageinindia.com), 6*, 7.

DéJean, H. (1998). Morphemes as necessary concepts for structures: Discovery from untagged corpora. In *Workshop on paradigms and grounding in natural language learning*, pp. 295–299.

Freitag, D. (2005). Morphology induction from term clusters. In *Proceedings of the ninth conference on computational natural language learning* (CoNLL), pp. 128–135.

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

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

Hafer, M. A., & Wess, S. F. (1974). Word segmentation by letter successor varities. *Information Storage and Retrieval, 10*, 371–385.

Harris, Z. (1955). From phoneme to morpheme. *Language, 31*(2), 190–222.

Keshava, S., & Pitler, E. (2006). A simpler, intuitive approach to morpheme induction. In *PASCAL challenge workshop on unsupervised segmentation of words into morphemes*.

Schone, P., & Jurafsky, D. (2001). Knowledge-free induction of inflectional morphologies. In *Proceedings of the second meeting of the NAACL*, pp. 183–191.

Singh, S., Gupta, K., Shrivastava, M., & Bhattacharyya, P. (2006). Morphological richness offsets resource demand – experiences in constructing a POS tagger for Hindi. In *Proceedings of the COLING/ACL 2006 poster sessions*, pp. 779–786.

Snover, M. G., & Brent, M. R. (2001). A Bayesian model for morpheme and paradigm identification. In *Proceedings of the 39th annual meeting of the ACL*, pp. 482–490.

Yarowsky, D., & Wicentowski, R. (2000). Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th annual meeting of the ACL*, pp. 207–216.