Statistical Stemming for Kannada

Suma Bhat
Beckman Institute for Advanced Science and Technology,
University of Illinois,
Urbana-Champaign, IL 61801, USA
spbhat2@illinois.edu

Abstract

Stemming is a process that groups morphologically related words into the same class and is widely used in information retrieval for improving recall rate. Here we study a set of statistical stemmers for Kannada, a resource-poor language with highly inflectional and agglutinative morphology. We compare stemming using simple truncation, clustering and an unsupervised morpheme segmentation algorithm on a sample from a text collection.

We observe that a distance measure that rewards longest prefix matches is the best performing clustering-based stemmer. However, using a reasonably performing unsupervised morpheme segmentation seems to outperform the other stemming schemes considered.

1 Introduction

With the ongoing quest for developing language processing techniques and tools for under-resourced languages there is an emerging need to study various aspects of these languages. Kannada, with nearly 70 million speakers is one of the 40 most spoken languages in the world. It is one of the scheduled languages of India and the official and administrative language of the state of Karnataka in South India. Its rich literary heritage has endowed the language with an immense written resource and efforts are currently underway to bring them to web scales. However, available computational tools for Kannada are only in their incipient stages. Simultaneously, there is an ever increasing number of internet users who are creating online materials in Kannada. As more information becomes available it becomes imperative to develop language processing tools that help us organize, search and understand information in Kannada. One such task is that of information retrieval and the time is ripe for developing language processing tools for efficient information retrieval in Kannada.

Stemming serves to conflate morphologically related word forms into a common form. The role of stemming in improving retrieval effectiveness, particularly for highly inflected languages and monolingual retrieval has been well documented in studies including (Larkey and Connell, 2003; Majumder et al., 2007; Dolamic and Savoy, 2010). In addition, the efficiency of IR systems owing to a decrease in size of the index term set (and a concomitant decrease in storage) is an added effect of stemming. Consequently, with the goal of developing a suitable stemmer for Kannada (a resource-poor language as far as current language processing resources are concerned), the focus of this study is statistical approaches to stemming in Kannada. We study stemming via three strategies. The first is truncation by a prefix of certain length, the second, a clustering approach using string distance measures and the third involves using the result of unsupervised morpheme segmentation. In this context the goals of the current study are the following.

1. To choose the most effective distance measure from among a set of string distance measures and determine a distance threshold that results in good stemming performance; and,

2. To compare the performance of the resulting clustering-based stemmer with traditional algorithms such as truncation and unsupervised morpheme segmentation.

For the purpose of this study, we evaluate the stemming performance without distinguishing between inflectional and derivational morphology.

The rest of this paper is organized as follows. In Section 2 a description in brief of prior research

The 4th Workshop on South and Southeast Asian NLP (WSSANLP), International Joint Conference on Natural Language Processing, pages 25–33,
Nagoya, Japan, 14-18 October 2013.

related to our study is presented. Section 3 deals with a description of the data used. The methodology used in the study is the content of Section 4. The experimental details are found in Section 5. Section 6 highlights the results and the related analysis occurs in Section 7. In Section 8 we present conclusions drawn from the study and outline some directions for further study.

2 Prior Work

As a recall enhancing technique in IR and efficient storage mechanism (since morphologically related word forms are conflated to the same stem), various stemming procedures have been studied for several languages. Stemming experiments and their effectiveness in IR have been carried out for English and other European languages (Goldsmith et al., 2000; Savoy, 2006; Fautsch and Savoy, 2009; Korenius et al., 2004; Majumder et al., 2008) and for a few Indian languages (Majumder et al., 2007; Dolamic and Savoy, 2010). There is empirical evidence that a stemming procedure improves retrieval effectiveness when applied to European languages including French, Portuguese, German and Hungarian (Savoy, 2006). For processing agglomerative languages such as Turkish and Finnish, stemming effectiveness has been studied (Ekmekçioglu and Willett, 2000; Sever and Bitirim, 2003; Korenius et al., 2004). Further, indexing and search strategies have been found to perform significantly better with the application of the various stemming strategies (compared to an indexing scheme without a stemmer), for Hindi, Bengali and Marathi (Dolamic and Savoy, 2010; Majumder et al., 2007). Retrieval experiments on English, French, Bengali, Hindi and Marathi datasets show that a statistical stemming approach via clustering is effective for languages that are primarily suffixing in nature (Majumder et al., 2007; Dolamic and Savoy, 2010; Ekmekçioglu and Willett, 2000; Sever and Bitirim, 2003).

As an effective bootstrapping approach to stemming, statistical methods have been explored for several languages. The popular stemming algorithms for English are based on language-specific rules explored in (Lovins, 1968) and (Porter, 1980), whereas for several other languages statistical stemming has been considered. The statistical stemmer studied in (Majumder et al., 2007; Šnajder and Bašic, 2009) is a cluster-based suffix stripping algorithm, whereas a probabilistic suffix stripping algorithm is explored in (Di Nunzio et al., 2004). In (Ekmekçioglu and Willett, 2000) a rule-based morphological analysis stemmer is used for Turkish.

Studies on developing morphological analyzers (mostly rule-based) for Kannada are available (Antony et al., 2010; Veerappan et al., 2011; Shastri, 2011; Murthy, 1999), but there have been no reports of evaluations of their propositions in terms of stemming effectiveness. In (Bhat, 2012), a preliminary study of unsupervised algorithms for morpheme segmentation in Kannada is available which observed that a statistical morpheme segmentation algorithm such as (Dasgupta and Ng, 2006) shows a reasonable performance for morpheme segmentation in Kannada. Taking the results of prior studies further, the current study is cast in the knowledge-base gained and compares stemming using a clustering-based approach with stemming using simple truncation and that using an unsupervised morpheme segmentation algorithm. The favorable results of prior studies with languages that are primarily suffixing in nature and those of the agglutinative type, prompts us to consider their stemming approaches for Kannada.

2.1 Challenges to Morphological Analysis in Kannada

Kannada is one of the four major literary languages of the Dravidian family. Kannada is mainly an agglutinating language of the strongly suffixing type (Dryer and Haspelmath, 2011; Sridhar, 1990). Words contain a basic root, with one or more suffixes being combined with this root in order to extend its meaning or to create other classes of words. Agglutination can result in long words that can contain as much semantic information as a whole English phrase, clause or sentence. An example of this characteristic is provided by the word, *sadupayOgapaDisikoLLabahudu* 'could avail the benefits'. Nouns are marked for number and case and verbs are marked, in most cases, for agreement with the subject in number, gender and person (like other nouns, geographical, product and even proper names are marked with case endings making the use of a dictionary impractical). This makes Kannada a relatively free word order language. Morphologically rich languages such as Kannada, are characterized by a large number of morphemes in a single word, where

morpheme boundaries are difficult to detect because they are fused together. In addition, rampant morphophonemic processes (sandhi), productive compounding and agglutinating morphology of inflectional and derivational suffixes (the latter mostly with words of Sanskrit origin naturalized into Kannada) drive the prolific word formation processes of the language (Sridhar, 1990).

3 Data

Corpus: For the purpose of experimentation we use the Kannada portion of the EMILLE corpus (Baker et al., 2002). In order to avoid words with typographical errors we restrict the sample to those word types occurring at least two times. Here, instead of resorting to clustering the lexicon from the given collection, we reduce the computational cost of the experiment by focusing on a subset of the lexicon. Thus, our experimental data set has 10020 words which are chosen so as to include the most morphologically productive words (manually chosen from the lexicon). We consider a suitable Roman transliteration of the Kannada unicode characters for computational ease.

Gold set: For evaluating the stemmers, we manually group the related forms of words in the sample into equivalence classes. In preparing the classes we do not distinguish between inflectionally and derivationally related forms. Thus, morphologically and semantically related words occur in the same group (for instance, compound forms semantically related to the root word occur in the same class). Moreover, in the case of polysemy, it suffices if some of their senses are related (note that semantic relations between members of such groups are less clear without the context). In such cases, the sense was arbitrarily chosen. It is worth pointing out that the groups here leave out compound words where the anchoring word occurs in the non-first position (a feature not intrinsic to Kannada, but derived from Sanskrit - e.g. *mahAraja* "great king" is derived from *raja* "king"). Our resulting gold set of equivalence classes consists of 667 such groups. Table 1 shows an excerpt from the sample in which 34 word forms occur in the same group.

4 Method

We perform a clustering-based approach to discover equivalence classes of root words and their morphological variants. Using a set of string distance measures a subset of a given text collection in Kannada is clustered to identify these equivalence classes. The proposed approach is compared with simple truncation based stemming and that based on unsupervised morpheme segmentation.

4.1 String Distance Measures

We consider the set of string distance measures proposed in (Majumder et al., 2007) for clustering a set of words. The distance measures assign a real value (distance) to a pair of strings, with the distance being indicative of the similarity between the two strings - a smaller value means higher similarity between the strings. Accordingly, we define the distance between two identical strings to be 0. For two strings X and Y, (if X and Y are of unequal length, we pad the shorter string with null characters to make the string lengths equal) let the length of the strings be $n+1$. Let m denote the position of the first mismatch between X and Y. The distance measures are given by,

$$D_2(X,Y) = \frac{1}{m}\sum_{i=m}^{n}\frac{1}{2^{i-m}},$$

$$D_3(X,Y) = \frac{n-m+1}{m}\sum_{i=m}^{n}\frac{1}{2^{i-m}},$$

$$D_4(X,Y) = \frac{n-m+1}{n+1}\sum_{i=m}^{n}\frac{1}{2^{i-m}}.$$

Intuitively, measure D_2 rewards long matching prefixes, D_4 penalizes long non-matching suffixes and D_3 does both. Motivated by the the effectiveness of the statistical stemmer (using the three measures) for the Indian languages - Hindi, Marathi and Bengali - as in (Majumder et al., 2007; Dolamic and Savoy, 2010), we use the same measures to study stemming for Kannada which is of the strongly suffixing type.

4.2 Clustering

We cluster the word forms using a hierarchical agglomerative algorithm as studied in (Majumder et al., 2007). The algorithm starts by assigning word forms to singleton clusters and proceeds by merging at each level the two least distant clusters until a single cluster remains. At each merging stage, the distance between two clusters is computed as one of the maximum, minimum, or average distance between the elements of the two

vidyArthi, vidyArthigU ,vidyArthigaLU , vidyArthigaLa, vidyArthigaLalli, vidyArthigaLalliruva, vidyArthigaLannU vidyArthigaLannu, vidyArthigaLeMdare,vidyArthigaLiMda, vidyArthigaLigAgi, vidyArthigaLigU, vidyArthigaLige, vidyArthigaLigiMta vidyArthigaLirabEku, vidyArthigaLoMdige, vidyArthigaLu, vidyArthige, vidyArthiniyara, vidyArthinilaya, vidyArthinilayada vidyArthinilayadalli, vidyArthinilayagaLu, vidyArthinilayakke, vidyArthiyAgi, vidyArthiyAgiddAga, vidyArthiyAgidda, vidyArthiyAgiddaru vidyArthiyAgidde, vidyArthiyU, vidyArthiya, vidyArthiyannu, vidyArthiyobbana, vidyArthiyu

Table 1: An example of a word group with words derived from *vidyArthi* "student"

clusters, referred to as complete-linkage, single-linkage, and average-linkage algorithm, respectively. Complete-linkage results in small and compact clusters, single-linkage results in long and branched clusters, whereas the result of average linkage is somewhere in between (in contrast to single linkage, each element needs to be relatively similar to all members of the other cluster, rather than to just one). For our experiment, we use average-linkage clustering. The resulting hierarchical structure is only used to the extent that members along a branch (below a certain distance) belong to the same cluster.

The main drawback of hierarchical agglomerative clustering is its computational inefficiency. Because the number of word forms in our sample is of the order of thousands, we adopt the following complexity reducing technique before proceeding to the hierarchical clustering step. We carry out the clustering in two consecutive steps: a divisive step followed by an agglomerative step. The idea is to use the divisive step to partition the set of word forms into pre-clusters and then to perform agglomerative clustering on each of the pre-clusters separately. In order to facilitate the merging of morphologically related words in the agglomerative step, it is essential that the pre-clusters be sufficiently coarse grained. We first group the words into pre-clusters by truncating them to the first n characters, $n = 1, \ldots, 7$. For a given prefix length, this results in a set of pre-clusters. This step serves a two-fold purpose in this study; it not only reduces the computational complexity, but also serves as a baseline for comparing the stemming performance of the subsequent hierarchical clustering step.

4.3 Unsupervised Morpheme Segmentation Algorithm

We consider an extension of Keshava and Pitler's algorithm (Keshava and Pitler, 2006) for language-independent morpheme induction studied in (Dasgupta and Ng, 2006). This algorithm was chosen since it was the overall best performing morpheme segmentation algorithm as studied in (Bhat, 2012) with a favorable F-measure of 72%. The algorithm (hereafter abbreviated as *Morphind*) being unsupervised, accepts as input an untagged corpus as a list of words with their frequency occurrence in the corpus. The key idea for morpheme discovery used here is that the conditional probability of a letter given its context exhibits a sudden jump at a morpheme boundary. The algorithm proceeds by first creating a forward tree and a backward tree, which provide the basis for efficient calculation of transitional probabilities of a letter given its context. It is then followed by the affix acquisition step, during which a set of morphemes is identified from the corpus. The third step uses the morphemes identified to segment words.

4.4 Evaluation

Since the explicit benefit of stemming is seen in the realm of information retrieval (IR), in several prior studies, the performance of a stemming algorithm is traditionally evaluated by considering the effect on the performance of IR systems. Conducting such a task-specific evaluation makes it impossible to assess cases where the stemmer makes faulty conflations and/or cases of correct conflation. In order to enable such an analysis we use a task-independent evaluation first proposed by Paice (1996). In this method of stemmer evaluation, the actual understemming and overstemming errors committed are counted on a manually constructed word sample in which the words are grouped based on their morphological relatedness. The understemming index (UI) is computed as the proportion of pairs from the sample that are relegated to different groups by the stemmer even though they are related; the overstemming index (OI) is computed as the proportion of pairs that actually belong to different groups among those that are conflated to the same stem. As in (Šnajder and Bašic, 2009) we resort to measure the overall stemming effectiveness in terms of stemming quality (SQ), defined as the harmonic mean of 1-

UI and 1-OI.

5 Experiments

5.1 Stemming Algorithms

We use *Morphind* with its default parameters (which govern the size of the suffix list for segmentation) as in (Bhat, 2012). The algorithm excludes from its analysis words seen only once in the corpus. This has the obvious disadvantage that several morphological variants are lost from being considered for the analysis.

5.2 Clustering

The distance threshold during the agglomerative clustering procedure governs the stemming quality; a lower threshold corresponds to 'light' stemming and a higher threshold corresponds to a heavier stemming. In (Majumder et al., 2007), the optimum distance threshold in the hierarchical clustering stage was chosen based upon the number of clusters being generated. As in (Šnajder and Bašic, 2009), we choose the optimum threshold for hierarchical clustering by looking at the stemming quality obtained by stemming the words in the gold set. The optimum threshold is chosen to be that which maximizes the stemming quality.

6 Results

Experiments were conducted using the sample of 10020 words and evaluated against the manually created equivalence classes of 667 groups.

6.1 Pre-clustering

Length	Total	Largest	UI	OI	SQ
1	26	1106	0.00	0.87	0.23
2	125	406	0.00	0.62	0.55
3	258	382	0.00	0.37	0.77
4	432	280	0.14	0.27	0.79
5	949	234	0.53	0.21	0.59
6	1840	108	0.73	0.03	0.42
7	2976	66	0.85	0.01	0.26

Table 2: Total number of clusters, size of the largest cluster, understemming (UI), overstemming (OI) indices and stemming quality (SQ) in the pre-clustering stage with different prefix lengths

The divisive step of clustering, with word forms truncated to the first n letters, $n = 1, \ldots, 7$, results in a set of clusters whose details are tabulated in Table 2. For each partition, we indicate the number of clusters, the size of the largest cluster, the understemming and overstemming indices and the stemming quality. The understemming index reflects the errors made by truncation (assigning morphologically related word forms to distinct pre-clusters by truncating to a prefix of given length). The problem of pre-clustering with a common fixed-length prefix is that, in order to obtain pre-clusters of manageable sizes, the prefix length must be high. This splits apart many morphologically related groups, as indicated by the increasing understemming values as we move down the column.

From Table 2, we notice that SQ peaks at a pre-clustering level of length 4. However, we also observe that the pre-clusters with a common prefix of length 3 have a lower understemming index. With the primary goal of keeping a low UI, and knowing that once split at the pre-clustering level, related words cannot be merged during the subsequent clustering stages, we choose pre-clusters with a common prefix of length 3 for subsequent agglomerative clustering. For hierarchical clustering, we use the distance measures D_2, D_3 and D_4 as defined in Section 4.1.

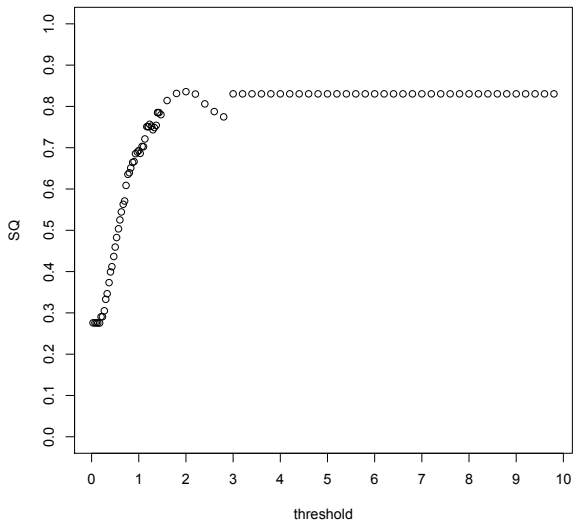

Figure 1: Plot of the stemming quality of clustering using the distance measure D_2 as a function of the distance threshold

In Figure 1, SQ of measure D_2 as a function of the threshold is shown. Measure D_2 achieves

the optimal stemming quality for a threshold of 2.0 and the maximum stemming quality is 83.46%.

Stemmer	t	UI%	OI%	SQ%
D_2	2.0	10.19	22.05	**83.46**
D_3	35.8	11.53	23.08	82.10
D_4	7.0	0.17	28.97	83.00
trunc(3)	-	0.25	36.69	77.20
Morphind	-	19.65	0.7	**88.82**

Table 3: Comparison of the optimal stemming performances of the approaches studied

Table 3 shows the optimal stemming quality of the string distance measures and the corresponding distance threshold values t. We note that clustering using measure D_2 yields the best performance for stemming by clustering. We notice that the optimum SQ, however, comes at the expense of a higher UI compared to the truncation scheme.

Next, comparing the stemming performance via clustering with that resulting from morpheme segmentation, we notice that the higher stemming quality (88.82%) in the latter case comes at the cost of a significantly higher UI.

The OI-UI plot on Figure 2 shows the stemming performance of the three distance measures. In particular, the OI-UI variation while varying the distance threshold is shown (since the distance thresholds were different for each measure, the corresponding information was omitted from the plot). As the value of the distance threshold increases, clusters are merged resulting in decreased UI and an increase in OI. The plot reveals that for the most part, the three measures perform better than simple truncation.

7 Discussion

An evaluation of the stemming performance crucially depends on the manually created groups of morphologically related word forms. Given that the grouping process was done by subjective human judgement rather than an objective criterion, variations in grouping are highly likely. For instance, semantic relatedness of words is different depending on the context and has varied granularity. To cite an example, consider the word group shown in Table 1. The group consists of words derived from *vidyArthi* "student". While *vidyArthigaLu* "students (nominative plural)" is related to *vidyArthi* by inflectional morphology, *vidyArthini* "female student" is also related and so is *vidyArthinilaya* " student residence" or *vidyArthivEtana* "financial aid (scholarship)". Depending on the topic being discussed, "student residence" and "scholarship" may be grouped together with "student" or not. We would like to remind the fact that words were grouped based only on their surface form with no additional information (such as context). It is important to also note here that in order to facilitate a clustering process, only words related by a suffixing process were grouped together which means that the manual grouping process has an inherent UI. As a result, words such as *BASe* "language" and *ADuBASe* "vernacular (common language)" were relegated to different groups.

With the choice of an appropriate distance threshold (for a description see Section 5.2), the SQ by clustering using measure D_2 can exceed 80%. Thus, clustering using string distance measures seems to be a favorable statistical stemming approach compared to simple truncation, which (on the same sample) reaches to SQ = 77.20%. The agreeable stemming quality of the truncation method suggests that a simplistic statistical stemmer, of taking fixed length substrings of words as stems, may in itself, result in an increased performance for IR.

The ease of implementation of the clustering procedure could be considered as an advantage over *Morphind*, which requires an informed setting of its parameters (language-dependent) for optimum segmentation performance. A more conclusive comparison of the two methods will have to be done in a task-specific setting such as that of IR evaluation.

Let us now take a look at an instance of the clustering procedure. We begin with the pre-cluster 'vid' (which included the group of words in Table 1) which was later clustered hierarchically. At the optimum threshold of t = 2.0, with the distance measure D_2, the group consisting of the morphological variants of *vidyArthi* was clustered along with the words found in Table 5. Here we notice that the words are broadly related, but certainly do not belong to the same group, indicating the source of understemming error.

We next look at the stemming (via morphological segmentation) of the set of words in the same Table 1 and notice that they have been divided into five groups in addition to the obvious first group, **vidyArthi**: there is **vidyArthiniyara**,

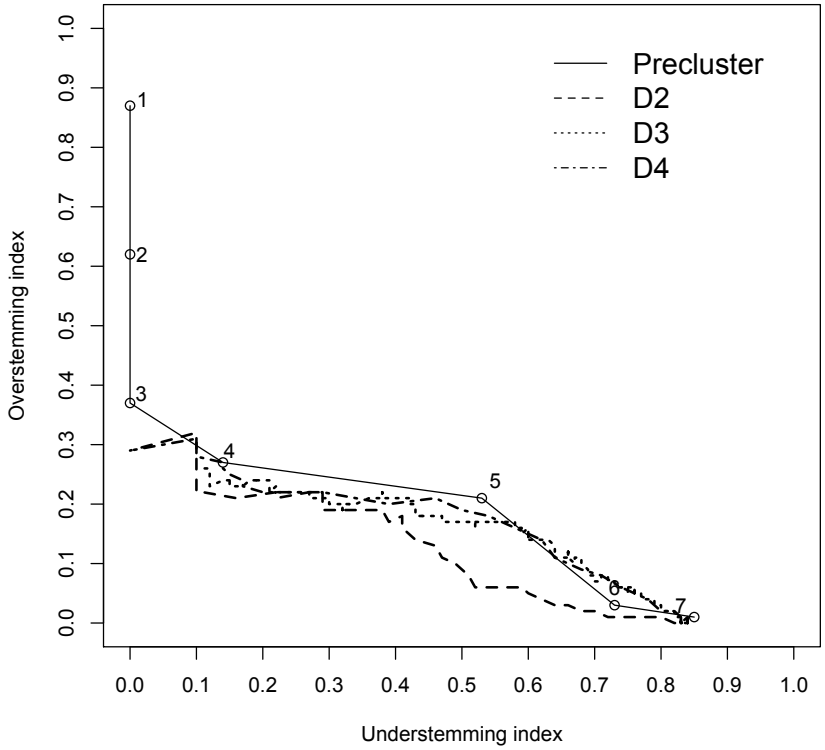

Figure 2: UI-OI plots of the hierarchical clustering with the distance measures compared with that in the pre-clustering with different prefix lengths

Method	No. of stems	Compression (%)
Gold set	666	93.00
Clustering(D2)	227	97.61
Truncation(3)	258	97.29
Morphind	1946	79.57

Table 4: Compression rates compared

vidyArthigaL, **vidyArthiyAgidde**, **vidyArthiyobbana** and **vidyArthinilaya**. The resulting overstemming error can be attributed to the inability of the morpheme segmentation algorithm to handle word forms such as *vidyArthinilaya* and *vidyArthiyAgidde* (compounding) and *vidyArthiyobbana* (external sandhi).

Another perspective of the stemming performance can be seen in Table 4. Here we list the compression rates in terms of the number of reduced words in the dictionary after stemming. The rather high compression ratio in the manually created grouping can be explained by noticing that the words comprise a subsample of the word types occurring in the collection chosen from among the most productive words occurring in the corpus. (True compression rates of the lexicon are expected to be lower than what is noted here, but certainly higher than that of English since the morphology is more complex than that of English.) Notice how the morpheme segmentation algorithm results in the creation of about three times as many groups as needed. This explains the corresponding high UI and low OI in Table 3

In the current study, the manually created equivalence classes did not distinguish between inflectional and derivational forms. If such a grouping were to be available, the distance threshold could have been tuned to isolate inflectionally related forms from derivationally related forms.

We are then interested to see how morphological variants are ranked based on their distance

vidyA2ara,vidyA2arara,vidyA2ikAri,vidyAByAsa,vidyAByAsada,vidyAByAsadalli,vidyAByAsakkAgi, vidyAByAsakke,vidyAByAsavannu,vidyAbud2i,vidyAbud2ivaMtaru,vidyAdAna,vidyAlayada,vidyAlayagaLalli, vidyAmaMtrigaLa,vidyAni2i,vidyAnilayada,vidyApariNitaru,vidyArthivEtana,vidyArthivEtanavannu,vidyAraNya, vidyAraNyara,vidyArhate,vidyArjaneyalli,vidyAsaMs4egaLalli,vidyAtajxara,vidyAvaMta,vidyAvaMtanAgalu, vidyAvaMtanU,vidyAvaMtarAdarU,vidyAvaMtarAgi,vidyAvaMtarU,vidyAvaMtara,vidyAvaMtaralli,vidyAvaMtarannAgi, vidyAvaMtarannu,vidyAvaMtarige,vidyAvaMtaru,vidye,vidyegU,vidyegaLa,vidyegaLannu, vidyeyAgide,vidyeya,vidyeyalli,vidyeyannU,vidyeyannu,vidyeyu

Table 5: The set of words clustered along with words derived from *vidyArthi* "student"

from the lemma (the dictionary form of the root). It would be convenient to have a distance measure that helps us distinguish between inflectional forms and derivational forms of a lemma. Towards this, we again consider the lemma *vidyArthi* "student" and its morphologically related forms in Table 1. We consider the distance between the lemma and the other forms using the measures D_2, D_3 and D_4 and rank order them with increasing order of distance (words with tied distances are grouped together and only the top-10 words are listed).

D2 = {**vidyArthi**, {**vidyArthigU**, **vidyArthige**, **vidyArthiyU**, **vidyArthiya**, **vidyArthiyu**}, {**vidyArthigaLU**, **vidyArthigaLa**, **vidyArthigaLu**, vidyArthiyAgi}, **vidyArthigaLalli** }

D3 = {**vidyArthi**, {**vidyArthigU**, **vidyArthige**, **vidyArthiyU**, **vidyArthiya**, **vidyArthiyu**}, {**vidyArthigaLU**, **vidyArthigaLa**, **vidyArthigaLu**, vidyArthiyAgi}, **vidyArthiyannu**}

D4 = {**vidyArthi**, {**vidyArthigU**, **vidyArthige**, **vidyArthiyU**, **vidyArthiya**, **vidyArthiyu**}, {**vidyArthigaLU**, **vidyArthigaLa**, **vidyArthigaLu**, vidyArthiyAgi}, **vidyArthiyannu**}

Here we observe that inflectionally related forms (in bold) are ranked higher than derivationally related forms. It is also worth pointing out that the stemming procedures considered here work by targeting the primarily suffixing processes of Kannada. As a result of this, forms related resulting via the umlauting process (such as *shikSaNa* becoming *shykSaNika* as a denominal adjective) are not conflated.

The stemming performance of *Morphind* is critically dependent upon the accuracy of the morpheme segmentation process. In (Bhat, 2012) an F-measure of 72% was reported. Upon closer inspection, we notice that several assumptions in the algorithm are specific to English, resulting in a low recall for Kannada, whose morphology is more complex than that of English. In particular, the assumptions that - all stems are valid words in the lexicon, affixes occur at the beginning or end of words only and affixation does not change stems - are clearly violated in Kannada. Moreover, as noted in (Bhat, 2012), morpheme segmentation is better for nouns than for verbs. An intuitive explanation here is that the noun inflectional process is seen more often than the verb variants.

8 Conclusions and Future Work

The results of the above experiment suggest that, for Kannada, the stemming performance of a truncation-based algorithm can be improved by subsequent clustering of the generated groups. Moreover a string distance measure, such as D_2, that rewards long matching prefixes emerges as an effective distance measure for clustering morphologically related words. Overall, comparing statistical stemmers, an unsupervised morpheme segmentation approach (such as *Morphind*) to stemming works better than a clustering-based approach in the case of Kannada.

We are well aware that a rule-based linguistic stemmer will perform better than a non-linguistic statistical stemmer. However, this exercise serves as a bootstrapping mechanism of the stemming process for Kannada. Looking ahead, the performance of an aggressive stemmer that not only removes inflectional suffixes but also removes derivational suffixes via a set of rules would be of interest.

This work does not account for possible word ambiguities since each word is taken as an isolated entity, and is grouped according to its typical connotation. As noted in (Paice, 1996), it would certainly be of interest to allow different senses of a word to be assigned to different groups, using information about the meaning of each specific word taken in its context.

Finally and most importantly, we plan to study stemming for Kannada in a task-specific setting such as that of IR evaluation as has been done in several related studies with other languages.

References

P. J. Antony, M.A. Kumar, and K. P. Soman. 2010. Paradigm based morphological analyzer for kannada language using machine learning approach. *International Journal on Advances in Computational Sciences and Technology ISSN*, pages 0973–6107.

Paul Baker, Andrew Hardie, Tony McEnery, Hamish Cunningham, and Robert Gaizauskas. 2002. Emille, a 67-million word corpus of indic languages: Data collection, mark-up and harmonisation. In *Proceedings of Language Resources and Evaluation Conference (LREC)*.

Suma Bhat. 2012. Morpheme segmentation for kannada standing on the shoulder of giants. In *Proceedings of the WSANLP*, pages 411–415.

Sajib Dasgupta and Vincent Ng. 2006. Unsupervised morphological parsing of bengali. *Language Resources and Evaluation*, 40(3):311–330.

Giorgio Di Nunzio, Nicola Ferro, Massimo Melucci, and Nicola Orio. 2004. Experiments to evaluate probabilistic models for automatic stemmer generation and query word translation. In *Comparative evaluation of multilingual information access systems*, pages 220–235. Springer.

Ljiljana Dolamic and Jacques Savoy. 2010. Comparative study of indexing and search strategies for the hindi, marathi, and bengali languages. *ACM Transactions on Asian Language Information Processing*, 9(3):1–24.

Matthew S. Dryer and Martin Haspelmath, editors. 2011. *The World Atlas of Language Structures Online*. Max Planck Digital Library, Munich, 2011 edition.

F Cuna Ekmekçioglu and Peter Willett. 2000. Effectiveness of stemming for turkish text retrieval. *PROGRAM-LONDON-ASLIB*, 34(2):195–200.

Claire Fautsch and Jacques Savoy. 2009. Algorithmic stemmers or morphological analysis? an evaluation. *Journal of the American Society for Information Science and Technology*, 60:16161624.

John Goldsmith, Derrick Higgins, and Svetlana Soglasnova. 2000. Automatic language-specific stemming in information retrieval. In *Proceedings of the Workshop on Cross-Language Evaluation Forum (CLEF)*, pages 273–284.

Samarth Keshava and Emily Pitler. 2006. A simpler, intuitive approach to morpheme induction. In *Proceedings of 2nd Pascal Challenges Workshop*, pages 31–35.

Tuomo Korenius, Jorma Laurikkala, Kalervo Järvelin, and Martti Juhola. 2004. Stemming and lemmatization in the clustering of finnish text documents. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, CIKM '04, pages 625–633.

Leah S. Larkey and Margaret E. Connell. 2003. Structured queries, language modeling, and relevance modeling in cross-language information retrieval. In *Information Processing and Management Special Issue on Cross Language Information Retrieval*.

Julie Lovins. 1968. *Development of a stemming algorithm*. MIT Information Processing Group, Electronic Systems Laboratory.

Prasenjit Majumder, Mandar Mitra, Swapan Parui, Gobinda Kole, Pabitra Mitra, and Kalyankumar Datta. 2007. Yass: Yet another suffix stripper. *ACM transactions on information systems (TOIS)*, 25(4):18.

Prasenjit Majumder, Mandar Mitra, and Dipasree Pal. 2008. Bulgarian, hungarian and czech stemming using yass. In *Advances in Multilingual and Multimodal Information Retrieval*, pages 49–56. Springer.

Kavi Narayana Murthy. 1999. A network and process model for morphological analysis/generation. In *ICOSAL-2, the Second International Conference on South Asian Languages, Punjabi University, Patiala, India*.

Chris Paice. 1996. Method for evaluation of stemming algorithms based on error counting. *Journal of the American Society for Information Science*, 47(8):632–649.

Martin Porter. 1980. An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137.

Jacques Savoy. 2006. Light stemming approaches for the french, portuguese, german and hungarian languages. In *Proceedings of the 2006 ACM symposium on Applied computing*, SAC '06, pages 1031–1035.

Hayri Sever and Yıltan Bitirim. 2003. Findstem: analysis and evaluation of a turkish stemming algorithm. In *String Processing and Information Retrieval*, pages 238–251. Springer.

G. Shastri. 2011. Kannada morphological analyser and generator using trie. *IJCSNS*, 11(1):112.

Jan Šnajder and B Dalbelo Bašic. 2009. String distance-based stemming of the highly inflected croatian language. In *Proceedings of the International Conference RANLP-2009. Borovets, Bulgaria: Association for Computational Linguistics*, pages 411–415.

S.N. Sridhar. 1990. *Kannada*. Routledge.

Ramasamy Veerappan, P. J. Antony, S. Saravanan, and K. P Soman. 2011. A rule based kannada morphological analyzer and generator using finite state transducer. *International Journal of Computer Applications*, 27(10):45–52.