Corpus based Statistical approach for Stemming Telugu

M.Santhosh Kumar, Kavi Narayana Murthy
Department of Computer and Information Sciences
University of Hyderabad, India
email: santhosh_ai@yahoo.co.in, knmuh@yahoo.com

Abstract. This paper is about Corpus based Statistical approach for Stemming Telugu. Telugu, a language of the Dravidian family, is spoken mainly in southern part of India and ranks second among Indian languages in terms of number of speakers. Telugu is a highly inflectional and agglutinating language providing one of the richest and challenging set of linguistic and statistical features. There are few languages in the world that match Telugu in this regard. NLP Applications such as IR and IE are often hard because of, among other things, the large number of morphological variants for any of given term. High performance morphological analyzers have eluded researchers for a long time. This is where stemming comes into picture. Stemming is a technique in which the variant forms of a word are reduced to a common form, thereby also enabling extraction of common suffixes. Thus, words which are different in surface form but have a common stem are conflated. Well known techniques of stemming for other languages including English are rule based or dictionary based. They aim at the removal of suffixes based on certain rules or dictionary look-up. For a language like Telugu it is not easy to build a rule based Stemmer because of the large number of morphological variations and unavailability of adequate lexical resources. In this paper we describe several corpus based statistical techniques we have developed for Stemming for Telugu, including n-grams, suffix-trees and HMMs. We have also developed syllabification rules and include some syllable statistics here.
Keywords:- syllabification, stemmer, n-grams, suffix tree, morphology, HMMs.

1 Introduction

Stemming is a technique in which the various forms of a word are converted to a common root. Stemming is a common language processing task in most information retrieval systems, where it is used to improve the ability to match query and document vocabulary. Both inflectional and derivational morphology lead to multiplication of word forms and stemmers are designed to handle these. We may use a linguistic approach, using prior knowledge of the morphology of the specific language, or a corpus based approach based on statistical principles using a text corpus in the given language. Rules based approaches can be

more effective depending on the quality of morphological analysis. However, a linguistic approach implies expert manual labor. Stemming algorithms based on statistical methods require little manual effort.

A number of stemming algorithms have been proposed in the literature. Most of them are rule based or dictionary based. It is very difficult to construct a rule based stemmer for a language like Telugu which is so rich in morphology. According to a study by an expert linguist[1] there can be up to 10 lakh different forms for a single Telugu verb. Dictionary based stemmers are also not possible for Telugu since such dictionaries are not available. In recent times, corpus based statistical approaches have emerged as promising alternatives to traditional linguistic approaches. While corpus based and statistical approaches to languages have been well established elsewhere in the world, India is still lagging behind. In this paper we describe several corpus based statistical techniques we have developed for Stemming for Telugu, including n-grams, suffix-trees and HMMs. We have also developed syllabification rules and include some syllable statistics here.

2 A Survey of Stemming Techniques

There are several stemming techniques proposed in the literature. Here we survey some of them.

2.1 Lovins Stemmer

The Lovins Stemmer is a single pass, context-sensitive, longest-match stemmer developed by Julie Beth Lovins of Massachusetts Institute of Technology in 1968. Lovins stemmer maintains a list of most frequent suffixes, 250 in number, and it removes the longest suffix ensuring that the stem is at least 3 characters long. It is quite unreliable and frequently fails to produce the correct stem.

2.2 Porter Stemmer

The Porter stemmer is a conflation stemmer developed by Martin Porter at the University of Cambridge in 1980 [1]. Porter stemmer is designed for English language. Porter stemmer is based on the idea that suffixes of words are mostly made up of a combination of smaller and simpler suffixes. It has 5 steps applying rules within each step. If a suffix rule matches a word, then the conditions attached to that rule are tested and the stem is obtained by removing the suffix. Porter stemmer is a linear step stemmer. The Porter Stemmer is readily available and is widely used.

[1] Personal Communication, G. Uma Maheshwara Rao, CALTS, University of Hyderabad

2.3 Dawson's Stemmer

The Dawson Stemmer was developed by J.L. Dawson of the Literary and Linguistics Computing Centre at Cambridge University. It is based upon the Lovins stemmer, extending the suffix list to 1200 suffixes. It keeps the longest match and single pass nature of the Lovins stemmer. It replaces recoding rules which were found to be unreliable, using instead, an extension of the partial matching procedure, also defined within the Lovins Paper.

2.4 Krovertz Stemmer

The Krovertz Stemmer was developed by Bob Krovertz, at the University of Massachusetts, in 1993[2]. It is based on the morphology. Krovertz stemmer effectively and accurately removes inflectional suffixes and then checks a dictionary.

2.5 Paice-Husk Stemmer

The Paice-Husk Stemmer was developed by Chris Paice at Lancaster University in the late 1980s and was originally implemented with assistance from Gareth Husk[15]. It is an iterative stemmer. It removes endings from the word in a finite no of steps. It uses a separate file which has different sections for each letter from the alphabet. The sections are ordered alphabetically. An index is built from the last letter of the the word. It specifies an ending which matches the last letter of the word. If any special condition for that rule are satisfied (Ex:rule is applicable if no other rules are as yet applied). Application of a rule should not shorten the word by more than specified length.

The ideas and techniques used in these stemmers are by and large not applicable to morphologically rich languages such as Telugu. Simple affix-stripping will not suffice as Telugu morphology involves complex morpho-phonemic changes.

3 The LERC-UoH Telugu Corpus

A corpus is a large and representative collection of language material stored in a computer processable form. Corpus provides the basic language data from which a variety of lexical resources can be generated. The Telugu corpus developed at the Language Engineering Research Centre (LERC), Department of Computer and Information Sciences, University of Hyderabad, India, referred to as LERC-UoH corpus here, adds up to nearly 39 Million words, perhaps one of the largest corpora for any Indian language today [13]. From this Telugu text corpora, a list of 33,20,920 different word forms with frequencies has been extracted. This list forms the basis of all of our experiments here. It may be observed that available printed dictionaries of Telugu contains 15,000 to 30,000 root words. We thus have some idea about the complex nature of Telugu morphology.

The LERC-UoH corpus is in ISCII[2] encoding. For convenience, we have mapped the word list extracted from this corpus into Roman notation using a tool developed by us here [12].

3.1 Corpus Analysis and Pre-Processing

Basis of Statistical Stemming Telugu is primarily a suffixing language - an inflected word starts with a root and may have several suffixes added to the right. Suffixation is not simple concatenation - complex morpho-phonemeic changes occur at the junctures. The main idea in stemming is to divide a given word form into a root and a single suffix-complex including all the suffixes. No attempt is made to analyze the internal structure of the suffix complex.

The premise of statistical stemming is that the best place to cut a word into a stem and (a combination of all) suffix(es) is the one that globally maximizes the probability of the root as also that of the (combined) suffix. There are various levels at which this can be done. One could consider words as sequences of symbols, each symbol encoded as a single byte. Byte-level analysis is completely inappropriate for Indian scripts [14]. Orthographic units in Indian scripts are called akSara-s [13]. It is really morphemes which combine to form full words according to morpho-phonemic rules of the language and akSara-s are also not appropriate units for morphology or stemming since akSara-s correspond to C*V syllables alone in Telugu. Only syllables - spoken units making up morphemes, can form the right basis for stemming in Telugu. In this work, rules for syllabification have been worked out for Telugu, tested and refined. All of our experiments are based on the assumption that words are sequences of syllables and morpheme boundaries coincide with syllable boundaries.

Syllabification Syllabification is the separation of the words into syllables. Syllables are unit of organization of sequence of speech sounds. It is typically made up of a syllable nucleus (most often a vowel) with optional initial and final margins (typically consonants). Syllables are considered as phonological building blocks of words. The written separation is usually marked by hyphen. An example of a Telugu word 'tina:lanukuMTunna:Du' is syllabified and is written as ti-na:-la-nu-kuM-Tun-na:-Du.

Rules of syllabification The following rules for syllabification in Telugu have been found in literature. Here C is a Consonant and V is a Vowel. Every syllable is centered at a vowel and surrounding consonants are split as per the following rules:
RULE 1: Initial and final consonants in a word go with the first and last vowel

[2] Indian Script Code for Information Interchange

respectively.
RULE 2: VCV: The C goes with the right vowel.
RULE 3: 2 or more Cs between Vs: First C goes to the left and the rest to right.
Note: Telugu words never show two or more vowels in sequence.

We have carried out a small syllabification study with native speakers of Telugu to validate these rules. Based on this study, we propose the following additional rules. Specified consonants followed by y, r, l or v are taken as single compound consonants as given below. Here (x,y)(a) is understood as 'x' or 'y' followed by 'a'
RULE 4 : (Any consonant except y,H,M)(y) is taken as single consonant.
RULE 5 : (Any consonant except y,r,l,L,L ,h',n,n ,N,N ,M,H)(r) is taken as a single consonant.
RULE 6 : (k,c,T,t,p,g,j,D,d,b,m,s',S,s)(l) is taken as a single consonant.
RULE 7 : (k,c,T,t,p,g,j,D,d,b,s',S,s,r)(v) is taken as a single consonant. With these rules, the Telugu word list was syllabified to obtain 34,644 distinct syllables.

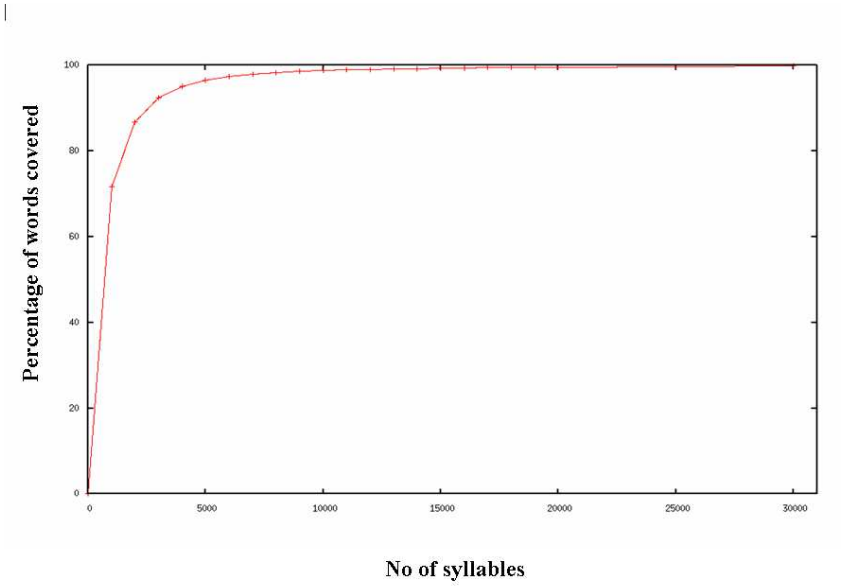

Fig. 1. Syllable Coverage Analysis

Coverage analysis Not all syllables occur with equal frequency. A coverage analysis is performed to explore the percentage of words covered by a given number of most frequent syllables. See figure 1. It may be observed that 5000 most frequent syllables account for 96.6% of the words in the whole corpus.

4 Proposed Stemming Techniques

4.1 Heuristic Stemmer

We initially explored a variety of heuristics under the premise that *"the best place to cut a word into a root and a suffix is the one that globally maximizes the probability of the root as also that of the suffix"*. After extensive experimentation, we have found the following heuristic score to give best results:
score = (2*P*S)/(P+S).
prefix score(P) = Frequency of prefix * length of prefix + 0.5.
suffix score(S) = Frequency of suffix * length of suffix + 0.5.
See results later.

4.2 N-gram based Stemmer

An n-gram is a substring of n consecutive tokens in a stream of tokens. Unigrams are n-grams with n=1, bigrams are n-grams with n=2 and trigrams are n-grams with n=3. Telugu is mainly a suffixing language and hence the initial portions of inflected and derived words generally match with the initial portions of the root word. Exceptions are rare. One exception is: 'ra:' and 'raMDi' are forms of 'vac'. Hence we could cluster all the word forms based on the word initial n-grams. Since mono-syllabic words are not many, we have clustered all the word forms based on the word initial bi-grams to obtain 2,67,502 clusters. The smallest word found within a cluster is taken as the root word or lemma.

4.3 Suffix Tree Approach

Here words are represented as a suffix tree. For each word and for each possible prefix, the *successive verity*[6] is calculated. Let p be a word of length l and p_i the first i characters of p. Let D be the set of words. $D(p_i)$ is defined as the subset of D containing words whose first i characters match p_i. The successive verity of p_i, denoted by $SV(p_i)$ is thus defined as the number of distinct characters (here syllables) which occupy the $i+1^{th}$ position in the words in $D(p_i)$". See figure 2. At the bottom of the figure *successive verity* for each possible prefix of the word *'tinna:nu'.* are given.

Stemming is performed at a position where the successive variety is maximum [6]. We observe that this criterion does not work well for Telugu. In many cases the maximum value occurs after the very first syllable. After extensive experimentation, we have obtained a set of heuristics to decide which among the first four maxima is to be taken for stemming.

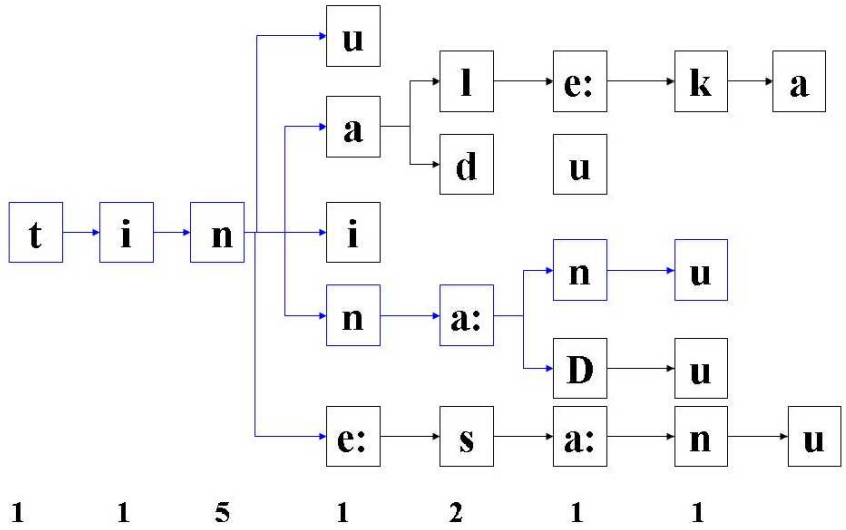

Fig. 2. An example of a Suffix Tree

4.4 Hidden Markov Models for handling External saMdhi

HMMs are finite-state automata which are doubly stochastic. State transitions are probabilistic functions and so are symbol emission from states. HMMs are called hidden because the states cannot be directly observed - what is observed are only the symbols they emit. A HMM is defined in terms of (i) the initial state probability matrix, (ii) the state transition probability matrix and (iii) the observation probability matrix.

HMMs are particularly useful in modeling processes that are in general unknown but that can be observed through a sequence of observed symbols. For instance, the sequence of letters that forms a word in a given language can be considered as a sequence of symbols emitted by a HMM. The HMM starts in a state that has non-zero probability of being an initial state, performs a sequence of transitions between its states emitting a new letter at each transition. In general, many state sequences or paths can correspond to a single word. It is possible to compute the probability of each path, and hence compute the most probable path that corresponds to a word. This problem is normally known as decoding, for which an efficient algorithm - the Viterbi algorithm, exists citeP10.

The states in our HMM correspond to positions in a word in terms of syllables. At every position there are two possibilities - the word continues into next syllable or the word ends. The topology of the HMM is accordingly fixed - from every state we have only two transitions, one to the next state and one back to the initial state (indicating the end of the current word and possibly beginning a new word). We restrict the number of states to 12 based on observations of word length in terms of syllables over the corpus. The observation probability matrix indicates the probability of observing a given syllable at a given position in a word. The initiate state probability matrix is defunct. The HMM parameters are estimated from the word list extracted from a large corpus.

This model can be used for identifying cases of external saMdhi - where two full words are conflated resulting in a single token. A transition to the initiate state is indicates a position where one word is complete and the next word in saMdhi begins. Viterbi algorithm is used to obtain the most likely state sequence for a given token and if a transition to initiate state is found in the optimal state sequence so obtained, the token is divided into two separate words at that point and each word stemmed in turn. Manual evaluation on a test data set of 300 words has given a Precision of 89% and Recall of 78%.

5 Performance measures

Usually the performance of a Stemmer is analyzed in terms of its contribution to performance improvement to an IR system. Here we use the following alternative measures of performance.

5.1 Accuracy

Accuracy is calculated manually over a test data set of 500 words randomly picked from the corpus.

5.2 Strength of a stemmer

Index Compression Factor (icf) icf represents the extent to which a collection of words is reduced by stemming.
icf = (N-S) / N where
N = Number of unique words before stemming and
S = Number of unique stems after stemming.

Number of Words per Conflation Class (wc) This is the average number of words reduced to a given stem. This metric is obviously dependent on the number of words processed, but for a word collection of given size, a higher

Table 1. Performance of various Stemmers proposed on a Data Set of 500 words

Type of Stemmer	wc	icf	% Accuracy
Heuristic	5.71	0.68	70.8
n-gram	12.41	0.69	65.4
Suffix Tree	2.83	0.51	74.5

value indicates greater reduction.
wc = N / S where
N = Number of unique words before Stemming
S = Number of unique stems after Stemming

It may be observed that greater reduction does not necessarily imply increased accuracy. In these experiments we find that the Suffix Tree approach to be the best in terms of accuracy of stemming.

6 Conclusion

In this paper we have proposed several corpus based statistical stemming techniques including heuristic based, n-gram based and suffix tree based techniques for Telugu. All the algorithms have been implemented in Perl under Linux. Experiments and results are included. In all cases, words are viewed as sequences of syllables and we have included syllabification rules which are an improvement over the standard rules found in literature. We have also shown how HMMs can be used to break external saMdhi.

We plan to explore combinations of these ideas to develop better stemmers. We also plan to use these techniques for developing high quality lexical resources including root words and suffix combinations. Bootstrapping techniques will be explored.

References

1. M.F.Porter, "An algorithm for suffix stripping", Program,14(3),130-137.
2. Krovertz, "Viewing morphology as an inference process", In proceedings of the 16th annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 191-202.
3. Mario A.Nascimento and Adriano C.R.da Cunha, "An Experiment Stemming Non-Traditional Text", SPIRE'98, Proceedings,p.75-80.Santa Cruz de La Sierra,Bolivia,Sep/98.
4. Massimo Melucci and Nicola Orio, "A Novel Method for Stemmer Generation Based on Hidden Markov Models", Proceedings of the 12th international conference on Information and Knowledge Management. ACM Press 131-138
5. http://202.41.85.117/ santosh556/cknm.txt
6. G.Bharadwaja Kumar, Kavi Narayana Murthy, B B Chaudhuri "Statistical Analysis of Telugu Text Corpora", To appear in IJDL., Vol 36, No 2, June 2007.

7. Kavi Narayana Murthy and G Bharadwaja Kumar, "Language Identification from Small Text Samples", Journal of Quantitative Linguistics, Vol 13, No. 1, 2006 pp. 57-80
8. Paice C, Husk G., "Another Stemmer", ACM SIGIR Forum 24(3): 566, 1990