LECTURE COMPENDIUM

simple'04

Symposium on Indian Morphology, Phonology & Language Engineering

Indian Institute of Technology, Kharagpur

19 – 21 March, 2004

Organizers

Dept. of Computer Science & Engineering,
IIT Kharagpur

Central Institute of Indian Languages
Manasagangotri, Mysore

Communication Empowerment Laboratory,
IIT Kharagpur

Lecture Compendium: SIMPLE'04 – Symposium on Indian Morphology, Phonology & Language Engineering

Published: 19th March 2004

Requests for copies may be mailed to:

Sudeshna Sarkar
Associate Professor,
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Dist. Midnapore, PIN: 721302
West Bengal, INDIA

Ph No. +91 (3222) 283494
email: sudeshna@cse.iitkgp.ernet.in

Printed by
Shyama Printing Works
Prembazar, Hijli Co-operative
Kharagpur – 721306
Ph No. +91 (3222) 277366

Preface from the Conveners

Computer Aided Processing of Natural Language is a rich research area with several interesting and open issues that have engaged the scientific minds all over for years. The area assumes a different dimension for multilingual India, where the rich repertoire of contents available in different languages, including English, needs to be made available to all, through alleviation of the language barrier. This need is specially pronounced for the rural people and the *pre* and *neo* literates, who will remain deprived of the available information, unless the barrier is overcome.

In India, several research groups are engaged in research in Indian Language Speech and Natural Language Processing. IIT Kharagpur is no exception. At the Communication Empowerment Laboratory, IIT Kharagpur, extensive research is being carried out on different aspects of Speech and Natural Language Processing in Indian Languages. A need that is commonly felt, is the necessity of roping in Computer Scientists and Linguists (Computational Linguists) under a common activity umbrella and establish collaboration, without which Indian Language NLP will not achieve the desired maturity. Moreover, Speech and NLP research is extremely resource intensive and requires the availability of good quality speech and text corpora in different languages and dialects. This is a Herculean task that again calls for collaboration among research groups and sharing of resources.

The objective of this symposium is to initiate such interactions among the linguists and computer scientists in India. We have made this event free from registration fee to enable new researchers to participate. We have been overwhelmed by the response that we obtained from every corner of India and we are very happy to have representations from almost all active research groups in India. It was indeed a difficult task to pack in the presentations in a span of two days and we had no option but to extend it by half a day. The entire schedule has been organized in six paper presentation sessions, two plenary sessions and two sessions for poster presentations and demonstration of working systems. The paper presentation sessions cover the issues of Morphology, Phonology and Speech, Machine Translation, Applications of Speech and Natural Language Processing, and Development of Lexical Resources along with Semantic Role Tagging.

We are extremely happy and fulfilled that several luminaries of linguistic research are participating in this symposium to share their ideas. We have received complete support from the Central Institute of Indian Languages (CIIL), Mysore, who are the joint organizers of this Symposium. CIIL Mysore has come forward with travel support for many of the contributing participants. We are also supported in this endeavour by the Natural Language Processing Association of India (NLPAI).

We are extremely grateful to the Ministry of Communication and Information Technology, Govt. of India and the Intel – *Innovation in Education* for extending financial support towards this event, as co-sponsors.

Anupam Basu
Sudeshna Sarkar

Conveners SIMPLE 04
Dept. of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

SIMPLE'04 Organization

Convenors:	Anupam Basu, Dept. of Computer Science & Engg. IIT Kharagpur Sudeshna Sarkar, Dept. of Computer Science & Engg., IIT Kharagpur
Steering Chair:	Sujoy Ghosh, Head, Dept. of Computer Science & Engg., IIT Kharagpur
Organizing Chair:	Arobinda Gupta, Dept. of Computer Science & Engg., IIT Kharagpur
Finance Chair:	Shamik Sural, School of Information Technology, IIT Kharagpur
Administrative Coordinator:	D. Gunasekaran, Registrar, IIT Kharagpur
Coordinator, CIIL:	B. Mallikarjun, Central Institute of Indian Languages, Mysore
Organizing Team:	Monojit Choudhury, Dept. of Computer Science & Engg., IIT Kharagpur Pradipta Ranjan Ray, Dept. of Computer Science & Engg., IIT Kharagpur Samit Bhattacharya, Dept. of Computer Science & Engg., IIT Kharagpur Samiran Sarkar, Dept. of Computer Science & Engg., IIT Kharagpur Plaban Kumar Bhowmik, Dept. of Computer Science & Engg., IIT Kharagpur Anirban Lahiri, Dept. of Computer Science & Engg., IIT Kharagpur Arijit Mukhopadhyay, Dept. of Computer Science & Engg., IIT Kharagpur Samit Patra, Communication Empowerment Laboratory, IIT Kharagpur Rennis Thomas, Communication Empowerment Laboratory, IIT Kharagpur Tamali Bhattacharya, Communication Empowerment Laboratory, IIT Kharagpur Mousumi Goswami, Communication Empowerment Laboratory, IIT Kharagpur Rahul Maitra, Dept. of Computer Science & Engg., IIT Kharagpur Dande Jagadish, Dept. of Computer Science & Engg., IIT Kharagpur Kaushik Chakrabarty, Dept. of Computer Science & Engg., IIT Kharagpur Prakash Mundra, Dept. of Computer Science & Engg., IIT Kharagpur Shiraj Sen, Dept. of Mathematics, IIT Kharagpur Sudipta Kundu, Dept. of Mathematics, IIT Kharagpur Abhishek, Dept. of Computer Science & Engg., IIT Kharagpur Shera Chakrabarty, Communication Empowerment Laboratory, IIT Kharagpur Sandipan Dandapat, Communication Empowerment Laboratory, IIT Kharagpur

SIMPLE'04 Invited Speakers

Chief Guest: **Dr. R. K. Arora**
Senior Director,
Ministry of Communication & Information Technology,
Government of India

Keynote: **Prof. Pabitra Sarkar**
Former Vice Chancellor, Rabindra Bharati University
Kolkata

Invited Talks: **Dr. Om Vikas**
Senior Director,
Ministry of Communication & Information Technology,
Government of India

Prof Uday Narayan Singh
Director,
Central Institute of Indian Languages, Mysore

Plenaries: **Prof B. N. Patnaik**
Dept. of Humanities & Social Sciences
Indian Institute of Technology, Kanpur

Prof Rajeev Sangal
Director,
International Institute of Information Technology,
Hyderabad

Prof Bidyut Baran Chaudhuri
Head, Computer Vision and Pattern recognition Unit
Indian Statistical Institute, Kolkata

Prof Gautam Sengupta
Jadavpur University, Kolkata

Special lectures: **Prof G. Uma Maheshwar Rao**
University of Hyderabad

Prof Amitabha Mukerjee
Dept. of Computer Science & Engineering
Indian Institute of Technology, Kanpur

Prof Asoke K. Datta
Indian Statistical Institute, Kolkata

Prof Pushpak Bhattacharyya
Dept. of Computer Science & Engineering
Indian Institute of Technology Bombay

SIMPLE'04 Sponsors

Ministry of Communication & Information Technology Government of India

SIMPLE'04 Collaborators

Indian Institute of Technology Kharagpur

Central Institute of Indian Languages, Mysore

NLP Association of India

Contents

Posters

Demonstrations

INVITED PAPERS

COMPUTATIONAL LINGUISTICS FOR INDIAN LANGUAGES

B.N.Patnaik
Department of Humanities and Social Sciences &
Centre for Creative Writing and Publication
Indian Institute of Technology Kanpur
Email: *patnaik@iitk.ac.in*

Description of our languages within the framework of computational linguistics is in itself a legitimate research objective because it meaningfully occupies the space between descriptions offered by grammars of the traditional type and those with restrictive theoretical commitments. The former often allows imprecision and shows lack of rigour, whereas the latter is rather quite narrow with respect to domain, a consequence of its goals which have to do with explanation rather than description. It is a plus point that the descriptions that the computational linguistic enterprise would generate can be put to use for the development of language technologies. With this broad perspective in mind, I here propose to draw attention to certain aspects of language which should receive considerable attention in computational linguistic studies of our languages.

At one level, the foremost of these is arguably speech. That language is basically speech and is writing only secondarily is a view that many hold; however, linguistic research has not been seriously concerned with speech. To the best of our knowledge, there isn't a speech corpus of any of our languages. Then almost all the grammatical works that we have on languages, including our own, are grammars of the written forms; so one might conclude that linguistic research has really worked under the assumption that the grammars of the spoken and the written forms are basically non-distinct. This is obviously not correct. There are constructions that may appear in writing, but not in speech; the so-called "garden path" construction, for example. The notion "grammatical" or "well formed", for instance, is not the same in both. Clarity or elegance, rather than grammaticality, may be a better indicator of effective speech, which does not hold for the written form. Half sentences, sequences of words that do not necessarily form legitimate chunks (constituents), even single words, copiously, in fact, characteristically, occur in speech. Elliptical constructions commonly occur in speech, and addresses, calls, responses and the like constitute the integral part of any conversational exchange. In addition, a particular response does not have the same value in all its occurrences in a conversational discourse. The Oriya response "haan" may mean "I understand what you are saying" in one of its occurrences, and "I agree with you" in another, and "I am somewhat skeptical about what you are saying", in the third. The Oriya entity "he" in one occurrence is a "call", in another, an address. "Naain" is a negation marker in this language, but as response it does not always express negation – one of its functions is that of a starter. Understanding speech critically involves understanding the semantics and the pragmatics of these and many other similar entities, and it is only to be expected that the tagging of a corpus of speech would include certainly semantic, and wherever possible, pragmatic tagging.

Apart from linguistic scholarship, careful study of speech is obviously a very basic requirement for developing speech – to - speech systems involving Indian languages, which is by no means an unreasonable goal. Among other resources, including computational lexicons, what are needed for this are comprehensive computational grammars of Indian languages. In the lack of a speech corpus of any of our languages, there is expectedly as yet no skeletal or partial grammar of the spoken form of any of these languages. Besides, there is no comprehensive grammar of the written form of any Indian language – if there was one, it could have formed the basis for the construction of a computational grammar. As expected, there is as yet no standard grammatical ("grammatical" is used here as a cover term to include all aspects of grammar: phonology, morphology, syntax, semantics and pragmatics) vocabulary which can be used for descriptive purposes. What further complicates matters is that the question of the choice of a model of description is one that awaits attention of the linguistic community: clearly it is not going to be one existing model or another, but one that would integrate the descriptive apparatuses of more

systems than one. Working out the details is by no means a trivial problem.

Turning to an area where grammar and lexicon merge, one identifies the so-called functional elements as deserving careful and detailed study - of their semantics and pragmatics. Among these are: adpositions, determiners, classifiers, a range of particles including the pragmatic particles, participles, the verb of the conjunct verb, the vector of the compound verb, for example. Owing to their lack of orientation towards communication, neither the existing grammars nor lexicons, with or without an explicit computational objective, deal with the semantics of these elements, although they carry considerable semantic load, and are key to the unpacking of the semantics of utterances. Translation systems would benefit enormously from a comparative study of these elements. Consider the semantics of "on" in the following: the show is on, the book is on the table, put on something nice for the party, and switch it on. Similarly consider "down" in the following: put it down, he rolled down the hill, and this will bring the prices down. There is no single entity in any of our languages that is the semantic equivalent of "on" or "down" in the above-mentioned expressions: it is necessary to know what in our languages carry the relevant semantic load. Consider now the Hindi vector "de" in "phenk diaa", "kar diaa", haraa diaa" and the way the information it yields in each is captured in English. There is no straightforward one-to-one equivalence. Clearly, working out the semantic equivalences of the functional entities, an enormously important exercise, is again by no means a trivial task.

Reduplicatives, lexical doublets, and echo compounds are characteristic features of our languages (rather than, say, English, which is why a comparative study of English and the Indian languages in this area is of great value), and these tend to occur far more frequently in the spoken discourse than in the written. This is what assigns special significance to them in the present context. Reduplicative is again an area where lexicon and grammar meet, whereas lexical doublet, echo compound, expressives, etc. constitute part of the lexicon. Although these are extremely interesting from the semantic (and pragmatic) point(s) of view, they still await detailed and careful study. Comparative study would yield rich information regarding semantic structure of languages, apart from being useful for translation systems, etc. In Oriya, the reduplication of the adjective in "saru saru chaaula", "hajaara hajaara barsa", "hasilaa hasilaa muhan" does not express meanings of just one kind. And reduplication is not confined to adjectives alone; verbs, nouns, etc. also reduplicate.

The submission here is that because of traditional lack of interest in the spoken form of language, and general neglect of semantics, crucial areas of grammar and lexicon have gotten neglected in linguistic research and that this needs to be remedied, especially if the computational linguistic community seriously intends to develop a meaningful speech-to-speech system, even in some non-immediate future. The paper identifies some areas which need prioritization in this effort.

.

MTeval: An evaluation methodology for Machine Translation Systems

Akshar Bharati , Rajni Moona, Smriti Singh, Rajeev Sangal, Dipti Misra Sharma
{*r_moona, smriti, sangal, dipti*}*@iiit.net*
Language Technologies Research Center,
International Institute of Information Technology, Hyderabad

Abstract

In this paper we present a methodology for evaluating multiple MT systems on the basis of comprehensibility and accuracy of translations. In this methodology, we use evaluation of the translations by human subjects, to rate the translations on a scale of 0 to 4. The evaluations are consistent even though a small amount of training is given to the evaluators. The cumulative score therefore is a good indicator of acceptability of translations across multiple machine translation systems under consideration. The results of this evaluation are used to do further diagnostic analysis by developers which is valuable in improving the system. In our implementation, we evaluated three machine translation systems to translate from English to Hindi over different test scenarios.

I. Introduction

Evaluation of a machine translation system is a subjective process. With the availability of a large number of machine translation systems, there arises a need for their methodological evaluation. Such evaluations can benefit the users as well as the developers of machine translation systems even when the concerns for the two are different. MT users need to know which system is appropriate for their specific requirements while the developers need a feedback to improve upon the heuristics. An evaluation tool can take into account various aspects such as implementation, practical application, comprehensibility and accuracy of translations. In this paper, we present a methodology to evaluate multiple MT systems using human beings as evaluators, taking their feedback and analyzing outputs using common statistical techniques. In our evaluation strategy, we also present a comprehensive acceptance score for each of the translation systems.

In this paper, we present the criterion and the evaluation procedure for evaluating the translation quality and acceptance of multiple MT systems. Our Approach typically includes the evaluation of the quality of the unedited translations on the basis of the following parameters: comprehensibility and accuracy. We deal only with sentence level translations. A single sentence, however long it may be, is treated as a single unit. Evaluation of MT systems can be performed to evaluate varied aspects and serve many purposes. We have a two-level evaluation as mentioned below.

(a) Adequacy evaluation: This determines the fitness of an MT system with respect to comprehensibility of translations.

(b) Diagnostic evaluation: This is to identify limitations, errors and deficiencies of the MT system. These may be taken care of by the researchers or developers.

II. Background Work

Several researchers have worked on evaluation techniques of machine translation systems and many measures and methods have been developed for this purpose. Attempts have been made to produce well designed and well founded evaluation schemes. SYSTRAN [1] and Logos have developed internal evaluation methods to compare results given by different versions of their own systems. Palmira Marrafa and Antonio Ribero [2] proposed quantitative metrics for evaluations based on the number of errors in an evaluation and the total number of possible errors. Rita Nüebel [3] presents a blueprint for a strictly user-driven approach to MT evaluation within a net-based MT scenario, which can also be adapted to developer-driven evaluations. The Van Slype report for the European Commission [4] provided a very thorough critical survey of evaluations done to date. Eagles Evaluation Group [5] also worked to establish standards in the field to come up with a theoretically sound framework for evaluation of a machine translation system. However, no consensus has ever been reached in defining one single evaluation procedure, applicable to a machine translation system in all circumstances.

An evaluation tool of machine translation enables one to evaluate an MT system in convenient manner. Unfortunately, the parameters for evaluation are numerous such as, implementation, practical application, comprehensibility, accuracy of translations etc. Further these parameters are subjective in nature often rendering the numerical metrics useless. In this paper, we present a simple model that is convenient to use and gives a reliable comparison of multiple MT systems.

A. *Brief Description*

To perform the task of evaluation we have developed an MT Evaluation Tool (MTeval). A sample of 30 sentences is taken by linguists from various sources (discussed in section III.B). Their translations are obtained from the MT systems to be evaluated. A set of evaluators is decided upon, who will rate the sentences (discussed in section III.A). These translations are made available to the evaluators in a random order to get an unbiased evaluation. The evaluators do not have a clue as to which translation is from which MT system. They judge each sentence on the basis of its comprehensibility. The target user here is a lay person who is interested only in the comprehensibility of translations. The evaluators evaluate the sentences on the basis of how successfully they can comprehend the translations. The translations may not be perfect but even if they are comprehensible, the scoring is done based on the degree of their comprehensibility. The evaluators give scores (varying from 0 to 4) to each translation according to the scoring scheme (discussed in detail in section III.D). On the basis of these scores, results are generated using general statistical techniques. A general acceptance percentage of each MT system is calculated using formula for simple average. Majority based score of number of acceptable sentences (out of 30 sentences) for each MT system is also provided to compare the results.

Based on these scores, error analysis of the systems' output is done. The sentences that have an average score of 4 are perfect translations and thus, need no analysis. The translations with scores 0, 1, 2 and 3 are analyzed. A team of linguists classify all the errors in terms of error-list prepared by them in advance. This analysis helps the team of developers to improve the performance of an MT system (discussed in detail in section IV.B). The rest of the paper discusses the evaluation methodology, a case study based on the methodology and conclusions for future work.

III. Evaluation Methodology

A. Selection of evaluators

Selection of evaluators is a complex issue and has to be judicially handled. Evaluators cannot be randomly selected. They can be broadly classified as the following.

Developers: All the team members of the MT system development team can evaluate the system. The team will have a combination of programmers, linguists, and other monolingual or bilingual people. They know exactly how the system is working, will be aware of plus and minus points of the system. Some of them may be too critical and others may be partial towards their own 'baby'. If they are the evaluators, the scores may not be very reliable.

Trained evaluators: Among the trained evaluators come the linguists, Experts in source language or target language and even people who have keen interest in analyzing languages. These may be too critical, and may not be able to stick to the general norms of evaluation specified by us.

Common bilingual users: Selecting a group of bilingual people who are not involved in any way with the development of the MT systems may give the most unbiased feedback. But it was seen that their scores vary a lot. That is mainly because there are so many aspects one may consider while evaluating an MT system. Each evaluator may give importance to a different aspect. Therefore, deciding on the aspect on which evaluation is to be done is a must and the evaluators must be explained its significance. A little training can be given in the beginning with a couple of examples but the actual subjective judgment lies with the evaluator.

Once the set of evaluators is chosen, and evaluation is over, we eliminate those evaluators who have done inconsistent scoring and are too strict or too liberal. Acceptability levels always vary from evaluator to evaluator. Gradually, after two or three evaluations, we can form a reliable set of evaluators.

The other issue is whether the set of evaluators should be a fixed one or changed with each evaluation. Having the same set of evaluators each time is not a good idea as they gradually become used to the translations or may get biased to any particular MT system. Choosing a new set each time will give varied results, which may not be reliable. Our approach, therefore, is to have a mixed set of evaluators with few permanent members and a majority of new ones.

B. Selection of set of sentences

There are several issues involved in the selection of set of sentences for a comprehensive evaluation. For example, the set could be constant, variable or a mixed one; the number of sentences may be small or large; the collection of sentences may be domain specific or generic.

The use of the sentences especially prepared, and identical from one evaluation to another makes it too easy to adapt a translation system to give excellent results on the standard sample. A constant set of sentences doesn't offer a chance to judge the efficiency of an MT system in handling all possible constructions. A variable set, on the other hand, doesn't help in finding out the improvement in the system for constructions that it couldn't handle earlier. In our approach, we chose the set of sentences that includes few sentences tried out previously along with the majority of fresh ones.

Although for the reasons of coverage, we would like to use a large number of sentences in the set, too many sentences bore the human evaluators and by the time they have evaluated a few sentences, the rest of the scores are not reliable. The number of sentences has therefore been fixed to 30 in our evaluation after considerable experimentation.

Input sentences are chosen randomly from newspapers, articles, reviews and people's day-to-day conversations. The set is carefully crafted so that not all sentences in the set belong to any specific domain. The sentences so chosen are mostly conversational.

Care is taken to ensure that sentences use a variety of constructs. All possible constructs including simple as well as complex ones are incorporated in the set. The sentence set also contains all types of sentences such as declarative, interrogative, imperative and exclamatory. Sentence length is not restricted although care is taken that single sentences do not become too long.

C. Scoring procedure

Before the evaluators start the evaluation, they are told that the only aspects to be considered are the comprehensibility and accuracy of the translations. The scores are given from 0 to 4 as per the level of comprehensibility (from incomprehensible to perfect translation).

Evaluators are asked to follow the following steps for evaluation.

- Read the target language (Hindi for example) translations first.
- Judge each sentence for its comprehensibility.
- Rate it on the scale 0 to 4.
- Read the original source language (English) sentence only to verify the faithfulness of the translation (only for reference).
- Not to read the source language sentence first.
- If the rating needs revision, change it to the new rating.

D. Scoring Scale

As mentioned earlier, the scores on the scale are designed to focus on comprehensibility followed by accuracy of translation. The scoring scheme is given in table 1.

0	Unacceptable (doesn't make sense)
1	Unacceptable (major errors in translation, comprehensibility seriously effected)
2	Acceptable (some errors in translation but comprehensible)
3	Acceptable (No major errors in translation, fully comprehensible)
4	Acceptable (Perfect translations)

Table 1: Scoring Scheme for MTeval

IV. Case Study

Three different English to Hindi machine translation systems were evaluated with MTeval. The evaluation was done at least once each month and whenever a new version of any machine translation system was released. Five evaluators were chosen each time. The evaluators were bilingual and had a good command over English as well as Hindi. The evaluation was performed through the web interface in which the original English sentences were not displayed by default, but the evaluators had an option of viewing the sentences whenever they wanted to. The evaluation results, over a span of 4 months with different set of sentences each time, are given in table 2.

	MT1	*MT2*	*MT3*
May 2003	3.33%	30.00%	83.33%
Jun 2003	16.00%	36.60%	73.60%
Jul 2003	6.00%	56.00%	53.00%
Aug 2003	5.00%	25.00%	40.83%

Table 2: Acceptance Percentage for the three systems as evaluated with MTeval

Results show that the Acceptance percentage varies from system to system in each evaluation. Acceptance Percentage of all the three systems drops in the month of August because the set of sentences chosen was comparatively difficult and complex to be handled by the systems.

MTeval tool also gives an absolute number of sentences which had acceptable translations (table 3). An acceptable translation is defined as the one that receives a rating between 2 and 4 by the evaluator. The table 3 gives the total number of acceptably translated sentences out of a maximum of 30 for each of the three machine translation systems MT1, MT2 and MT3.

	MT1	*MT2*	*MT3*
May 2003	7	14	26
Jun 2003	7	19	23
Jul 2003	6	21	21
Aug 2003	1	10	18

Table 3: Majority based acceptance score for the three systems evaluated by MTeval

An extract from a scoresheet on MTeval is reproduced in table 4. In the score-sheet, it is clear that although the scores vary from one evaluator to another there is a consensus of sorts. Therefore, the average scores were used to calculate the acceptance percentage.

A. Feedback from evaluators

We have a system of continuously enhancing MTeval by taking feedback from the evaluators after each evaluation. The feedback form contains several questions. Some of these questions are given below.

- Rate the evaluation tool in terms of its user-friendliness.
- How often do you find the need to view the sentence in source language for a given translation?
- Does it make any difference to the score after viewing the English sentence for a given translation?
- Is there a need to change the scoring scale?

We have made use of this feedback to enhance the user-friendliness of the MTeval. It has also helped us to make the evaluation process simple wherein even a non-expert can evaluate machine translation systems.

	Sentences	*Evaluators*					*Avg.*
		1	*2*	*3*	*4*	*5*	
English sent: The book was presented to me by the President							
MT1	vah pustak thaa priijent'ad' kii or mujhako samiip vah raasht'rapati.	1	1	0	0	0	0.4
MT2	Pustaka ne raashht'rapati paasa men' mujhako diyaa gayaa thaa	2	1	1	1	2	1.4
MT3	Pustaka sabhaapati se mere paasa upahaara diyaa gayii thii.	3	1	3	3	2	2.4
English sent: He moved the basket with the rod							
MT 1	vah muuvd' vah d'aliyaa ke saath vah chhadd~ .	1	0	1	0	0	0.4
MT2	Usane pan'kti shrrxn'khale d'aliyaa chalii .	1	1	0	0	1	0.6
MT3	usane chhadr~a se t'okarii hilaaii	4	3	4	4	4	3.8

Table 4: Score Sheet

B. Error Analysis

As mentioned in the beginning, we use the results given by MTeval to do diagnostic evaluation as well. The types of errors looked for in the translations are listed in the table 5. Two sets of translated sentences from MT2 were evaluated and then analyzed. The number of sentences in each set was 30. All the errors in the translated sentences were identified and their frequencies were noted. The table 5 shows the error frequency of the two sets of sentences as case study 1 and 2.

Error list	*CaseStudy*	
	1	*2*
Total Bilingual-Dictionary errors	6	12
Phrasal dictionary Errors	7	2
Agreement & Word-form gen	6	7
No rules to parse	6	3
Rule failure	2	3
WSD	3	12
Reordering	3	8
TAM	3	4
Negatives/Interrogatives	1	1
Chunker Errors	6	1
Vibhakti Errors	2	1
Repetition	3	1
Punctuation Errors	2	2
Substitution	3	1

Table 5: Error frequencies for two sets of sentences as translated by MT2

The total numbers of dictionary errors in the two sets are 13 and 14 respectively. It implies that

the resource developers of MT2 need to enhance the dictionaries used by their system. The errors related to agreement and word form generation show an increase from 6 to 7. The errors related to WSD (word sense disambiguation) have also shown a steep increase. This alerted the developers of MT2 to carefully go through their respective modules and modify their programs or to change their approach altogether. This exercise of adequacy evaluation followed by diagnostic evaluation has helped the development team of MT1 and MT2 to improve their systems.

V. Conclusion

We have used this approach in our tool MTeval to evaluate multiple machine translation systems, a number of times. We believe that this is the right, trustworthy and simple way to test the translation quality of an MT system. We do an elaborate error analysis to improve the MT system. At the moment, we have restricted ourselves to sentence level evaluation. In the next stage, we are planning to extend this approach to text level evaluation on the basis of its comprehensibility, coherence and style.

The MT2 system produces output in several target languages. It is proposed in future to evaluate the output of these different language translations, and do a comparative study of the output. We suspect that this will yield a comparative analysis of lexical resources such as dictionaries and a contrastive analysis of the languages.

References

[1] Van Slype,G., "Systran:Evaluation of the 1978 version of the Systran English-French automatic system of the commission of the European communities", The Incorporated Linguist , 18, 1979, pp.86-89.

[2] Marrafa, Palmira and Antonio Ribeiro "Quantitative Evaluation of Machine Translation Systems: Sentence Level", Proceedings of MT Summit VIII Fourth ISLE workshop 2001, Spain, pp. 39-43.

[3] Nuebel, Rita "MT Evaluation in Research and Industry: Two Case Studies", in proceedings of 14th Twente Workshop on Language Technology in Multimedia Information Retrieval, December 1998, University of Twente, The Netherlands.

[4] Van Slype, G. "Critical Methods for Evaluating the Quality of Machine Translation (Final Report)", prepared for the Commission of the European committees, Brussels.

[5] EAGLES, Expert Advisory Group on Language Engineering, "Evaluation of Natural Language Processing Systems (Final Report)", prepared for DG XIII of the European Commission, 1996.

MORPHOLOGY

A Generic Architecture for Morphological Generators of Morphologically Complex Agglutinative Languages

Uma Maheshwar Rao, G. , Chaithra, T.P. , Santosh Jena
University of Hyderabad
guraosh@uohyd.ernet.in/guraohyd@yahoo.com

1. INTRODUCTION

The quest for generic solutions for typologically similar languages in an efficient way for the generation of word forms is largely unfulfilled. The tasks involving morphological generation can be simple or complex depending on the language's morphological typology. A morphological generator is expected to generate all and only the morphologically well-formed word forms from an input consisting of one or more of morphological elements of the language. The main problem involved is the mapping between the canonical shape of the surface word forms with the sequences of morphological elements given as input. In other words, allomorphy plays a crucial role in the building of a generator. Determining the right choice of the given allomorph in the context requires an optimal and efficient computational implementation. The paper discusses the design and implementation of a generic system for synthesizing word forms considering the needs of flexibility of the design and the simplicity of computation that permit to plug-in language specific databases with minimum changes.

In the last decade, Natural Language processing of Indian Languages has witnessed a surge in the practical applications involving computational morphology. Agglutinative languages, theoretically speaking, present rich but simple morphology, since the ratio between the number of morphemes and the number of morphological categories (features) that they signal is close to one. However, many modern Indian languages, though agglutinative in their morphology in the usual sense of the term, present a highly complex morphology in their surface realization. Any computational analysis of these languages requires a deeper understanding and analysis of certain facts about the design features of their morphology. In agglutination, as long as the correspondence between the forms and the formatives (morphological features) remain constant, the long chains of so-called complex word forms do not really offer morphological complexity. However, when languages introduce allomorphy the entire perception drastically changes. Allomorphy not only tends to erase boundaries between the morphemes but introduce the concept of variability into the morphology, making thus the agglutinative morphology a complex phenomenon. Any computational architecture must have a mechanism in its design feature that deals with this kind of morphology. Here we propose an architecture for computer generation of complex word forms of complex agglutinative languages. This is designed to produce complex word forms which are licensed by the well-formedness conditions of that particular language. Motivations for this design are drawn from the desire for the simplicity of computational implementation and cross linguistic adaptation.

2. THE DESIGN

We propose to design a modular model that is essentially based on a hybrid approach that combines the two basic and primary concepts of analyzing word forms of terms of Word and Paradigm model and the basic allomorph or the underlying form of the Item and Process model respectively. The morphological model underlying this description is influenced by the needs of flexibility of the design that permits to plug-in language specific databases with minimum changes and the simplicity of computation. The problem of generating well-formed surface word forms from the corresponding underlying root/stem plus one or more affixes in morphologically complex agglutinating languages has now become comparatively easier than before.

The following are the components in the generic architecture of the generator proposed here:

1. A lexicon consisting of simple roots/stems in the language. Each entry is provided with the category information and inflection-type or paradigm type feature information.
2. A list of the categories of inflection or affixes (for ex. Tense, Aspect and Modal in the case of verbs and number, case and post-positions in

case of nouns) are organized in terms of layers or strata.
3. A module consisting of Inflectional GNP endings and Particles and clitics.
4. A set of rules to concatenate the roots/stems and affixes listed in 1., 2. and 3.
5. A set of tables projecting allomorphic variants. Each table constitutes a finite number of cells predetermined by the surface realization of the lexeme in the context of a morphological category.

The proposed architecture involves the identification of different strata or layers among the affixes, which enter into concatenation to generate word forms. It allows to modify and increment the system easily, since the suffixes are arranged in various layers or strata and separated from the main program. Unlike in traditional Item and Arrangement model, allomorphic variants are not given any special status if they are generable through simple morphophonemic rules. Similarly wherever required automatic phonological rules may be called for to derive surface forms.

3. THE IMPLEMENTATION

In the following, we describe in detail the organization of the data and the implementation of the generator. We illustrate the implementation of the generator for Telugu in terms of the following components:

3.1 The Lexicon

It is a list of root/stem words, each of these is marked for its category and inflexion-type membership or the paradigm type. These two features provide very important and crucial information in the generation of word forms satisfying the well-formedness conditions laid down by the morpheme structure conditions of the language.

3.2 Categories of TAM (Tense, Aspect and Modal) Inflection

Affixes of various inflectional categories are arranged into distinct sub-modules which are accessed from the category information provided in the lexicon with each entry. Members of these inflectional categories or affixes are segregated into distinct strata or layers based on the order of affixes in relation to the root/stem reflecting the constituency of the word forms in terms of primary inflection and followed by the secondary inflection. The inflection module is followed by two other modules viz. the Stem extension module and the Clitic module. The stem extension module involves three sub-modules, the AUX (auxiliary verb in the case of verbs) or Adverb (in the case of nouns); the PAR module stores a series of lists of particles which are highly specific to the given morphological category; and the GNP module which consists of a number of lists of gender, number person markings sensitive to the morphology of the host.

3.3 Inventory of Primary Inflection

The inventory of primary inflection stores language dependent data as a simple listing of all the possible affixes occurring in the primary inflection. Each of these elements is followed by a corresponding name or a label in English with a prefix hash. A suffix allomorph of suppletive type is also provided with a distinct slot as in 4. A#pt and 5. iM#pt in depicting morphologically conditioned allomorphs of past tense to avoid an ad hoc rule that converts A --> iM/_[pt]3ps nm.

3.4 Inventory of Secondary Inflection or GNP (Gender, Number, Person Endings)

GNP involves a series of pronominal endings grouped into distinct categories depending on their distribution. These lists include gender, number and person endings occurring on finite and nonfinite verbs, pronominalized nouns, adjectives, and adverbial nouns. A hash (#) separates comments from the actual affixes.

3.5 Inventory of PAR (Particles)

Particles are sub-words, which occur as appendages to major lexical categories following inflection. They are otherwise known as stem extensional elements with morpho-syntactic consequences. A number of such particles are identified and categorized according to the specificity of their distribution. The major difference between the clitics and particles is that the former can occur with any of the major lexical categories whereas the latter are specific to a particular category of inflexion, i.e. they are sensitive to the morphology of the host. The distribution of the mutually exclusive members of these lists is characterized by the host's primary inflectional category which is captured here by a concatenation rule.

3.6 Inventory of Clitics

A set of morpho-syntactically relevant sub-words are identified and listed. In Telugu there are two such groups of such clitics. The grouping of these clitics is based on their distribution and function. Clitics are generally external to any inflection. They generally do not exhibit preferences for a particular morphological category unlike the particles.

3.7 Inventory of AUX

It includes a number of auxiliary verbs or adverbials grouped into various lists specific to a particular inflectional category. Every time an AUX/ADV is selected it must be routed through the lexicon to get information with regard to its paradigm membership. This module is necessary to generate a large number of productive complex verbs/nouns indicating various morpho-syntactic functions.

3.8 Rules for Concatenation

A set of unordered rules to concatenate root and affixes of different strata are specified. Each such rule specifies the order of affixes and/or extensional elements to be selected and concatenated to the root. The following is a near exhaustive set of concatenation rules required for Telugu wordform generation. Numbers enclosed in parentheses but separated by a comma indicate alternate possibilities. A plus sign indicates obligatory concatenation. A hash precedes the concatenated wordform for exemplification.

3.8.1 Allomorphy. The allomorphy that is so frequently encountered and which is often paradigmatically determined (but cannot be dealt with by regular phonological rules) is dealt with here using simple delete and add tables derived from the generic forms or keywords of the paradigm types. Whenever patterns of allomorphy are beyond the range of phonological component, the paradigmatic information is called in to obtain the correct allomorph in the context. Depending upon the extent of it one may setup allomorphy tables for stems and affixes separately. There are different allomorphy tables for different category of words and are accessed independent of each other. A cell in each table deriving the relevant allomorph corresponds to intersection of the paradigm type and affix type which represent different rows and columns. A cell in the intersection of a row and a column predicts the correct allomorph (see, word-form generation tables, 1-6).

4. TESTING AND EVALUATION

Corpus-based methods require that there be a defined evaluation metric that produces a result, so that we can compare strategies or rule sets. If we cannot make this comparison, we have no reliable way of making progress, because we do not know which technique yields better result. The evaluation metric should have relevance to the task we are trying to perform, which helps the performance of the system to improve. Furthermore it is imperative and informative that new data is used for evaluation, to ensure that we have created a system that is robust with respect to the kind of data we expect the system to have to handle.

In the present model of morphological generator, the corpus is used to extract fully inflected word forms. Testing is done by marking the morpheme boundaries, running them on the generator, and then verifying the fully inflected output forms produced by the generator with the word forms extracted from the corpus. The testing process requires identification of the root forms, paradigm type and the exact inflections on the word. In case of a mismatch, the program reveals where exactly there is a mismatch, which could either be a problem in word form analysis or a problem with the design of the generator.

5. CONCLUSION

Morphological generation of complex word forms in complex agglutinative languages is not an easy task. Often these word forms involve multiple stems and multiples affixes. The task is not as simple as a straightforward generation by selection and concatenation. It involves a great deal of taming allomorphism, which requires a special module where the concatenated constructs get well-formed before they emerge as valid word forms. The proposed architecture can be used to build generators for the languages with more or less similar morphological complexity. Based on this architecture a Kannada wordform generator has already been implemented at CALTS and generators for Tamil, Oriya and Bangla are being implemented now.

REFERENCE

[1] Uma Maheshwar Rao, G., Materials for a Computational Grammar of Telugu: Vol.1: Morphology, CALTS: Univ. of Hyderabad (mimeo), 2003.

[2] Chaithra, T.P, A Morphological Generator for Kannada, CALTS: Univ. of Hyderabad, (M.Phil dissertation), 2003.

Bengali Derivational Morphological Analysis using Finite State Methods

Pradipta Ranjan Ray, Anupam Basu, Sudeshna Sarkar
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur 721302
e-mail: *{pradipta ,anupam, sudeshna}@cse.iitkgp.ernet.in*

Abstract

The paper presents a computational model which can be used for Bengali inflectional morphological analysis, and for exploring the feasibility of computationally modelling Bengali derivational morphology. The computational model is equipped to analyze both linear and nonlinear morphology, as they appear in Bengali, using finite-state methods. The analyzer would be useful in detecting the meaning of new words encountered during natural language understanding, grapheme to phoneme conversion, as well as probing the limits of computability of morphological analysis in Bengali.

1. Introduction

Morphological analysis plays an important role in natural language understanding by extracting information from the surface form of a word [**1**]. Morphology has been traditionally categorized into inflectional and derivational morphology, depending on the amount of change in the meaning of the word from that of the free morpheme. While finite state methods have been traditionally used for tackling inflectional morphological analysis and morpho-phonological problems [**2**,**3**,**4**], derivational morphology is rarely tackled using computational means [**5**,**6**,**7**].

Derivational morphology is very useful for guessing meanings and part of speech of unknown words [6]. In the context of Indo-Aryan languages, grapheme to phoneme mapping requires derivational morphological analysis [8]. Finally, computational modelling of derivational morphology helps us gauge the computational nature of word production (morphological synthesis) in Bengali [**7**].

In this paper, we present a computational framework for inflectional and derivational morphological analysis of Bengali. We use finite state methods in a framework which facilitates constraint propagation and allows nonlinear morphological analysis. Bengali, when written in the roman script is written using the ITRANS standard for transcription [**9**]. All text in ITRANS notation is italicized.

The paper is outlined as follows: the second section describes the aspects of Bengali morphology we are capturing in our morphological analyzer, the third section describes our computational model, and the fourth section is a summary based on our future work.

2. Aspects of Bengali Morphology

Bengali is an agglutinative, inflectional language of the Indo-Aryan family. Hence, the Bengali vocabulary is very large. The CIIL Corpus, consisting of 3,019,565 words contains as many as 182,848 distinct words [**10**].

The rate of growth of the vocabulary V(N) as a function of the size of corpus N has been studied in the literature [**11**] based on the Zipf and Mandelbrot-Zipf laws [**12**,**13**] . For inflectional and agglutinative languages, language specific parameters have values such that V(N) increases more rapidly with N (see Fig. 1). This implies that for morphologically rich languages, a powerful morphological analyzer is required as exhaustive enumeration of all words in the dictionary is unfeasible.

The different aspects of Bengali morphology that we are handling are as follows:

- **Noun inflections:** These morphological rules are linear in nature and more than four inflections are never concatenated. (eg. *rAmeraTAkeo* → *rAma + era+ TA + ke + o*)
- **Verb inflections:** These morphological rules are mostly linear, but may be nonlinear with respect to the penultimate syllable of the verb root. (eg. *herechhilAma* → *hAra + echhilAma*)
- **Noun prefixes:** *upasargas* or noun prefix rules are concatenative in nature and hence easily identified. (eg. *anidrA* → *a + nidrA*)

- **Compound word formation:** Rules for agglutination are linear in nature but may morph the morpheme boundary significantly based on *sandhi* rules. (eg. *dik* + *anta* → *diganta*, *rAta* + *dina* → *rAtadina*)
- **Verb and noun derivatives:** These rules (*pratyayas*) are often nonlinear and even irregular. (eg. *kaunteYa* → *kuntI* + *eYa*)

These cover most of the issues arising in Bengali morphological analysis. [**14**, **15**]

3. The Computational Model, Algorithm and Complexity Issues

The lexical resources required for the morphological analyzer and their abstract representations are as follows:

- The list of possible root words and affixes are stored as a finite state automata – the *lexicon*.
- The list of affixes which produce non-linearity (e.g. *ika* as in *dainika*) is not kept in the lexicon but stored as another finite state automaton – the *nonlinear affix lexicon*.
- The list of all possible sequences which may occur as a result of Bengali *sandhi* is stored as a finite state automaton – the *match automaton*. The null string is accepted by the *match automaton* for handling affixes and clean joining of two or more words (e.g. Bengali compound words like *rAtadina*).

We are deliberately not following each and every linguistic cue while modelling Bengali morphology in order to make our system faster and more accurate. The following deviations are notable:

- Unproductive affixes like *la* (eg *shItala*), *sha* (eg *karkasha*), irregular sandhi rules like *hins* + *a* → *siMha*, and unproductive *sandhi* rules like *t* + *D* → *DD* (eg. *uDDIna*) are being ignored and the generated words kept as root words.
- *sandhi* and *pratyaya* rules where the surface forms differ significantly from the underlying forms have been modified to make the underlying forms computationally easy to obtain, like *kShi* + *iSh~Nu* → *kShaYiSh~Nu* would be modelled as *kShaY* + *iSh~Nu* → *kShaYiSh~Nu* in keeping with *sah* + *iSh~Nu* → *sahiSh~Nu.*

Nonlinearity which occurs at a position a fixed distance away from the morpheme boundary, as in the case of Bengali verb morphology, is being modelled as linear morphology by including all

3.1 Algorithm for Morphological Analysis

Any surface form is analyzed as follows:

1. Represent the word as a word graph:
   - The word is represented in the graphemic form, and structured as a *word graph* $\mathbf{W_G}$. The word graph is a graph, where a traversal from any source node (indegree = 0) to any sink node (outdegree = 0) represents a possible morphological derivation.
   - $\mathbf{W_G}$ is initialized as a directed list with each grapheme at a node as in Fig. 2(a). The word may be analyzed from left to right or right to left, but in general Bengali *sandhi* rules have a wide range of right contexts, so right to left is preferable. The edges are from every grapheme to the grapheme immediately succeeding it.
2. Set *current grapheme* = rightmost grapheme
3. The neighbourhood of the current grapheme is inspected for possible morpheme boundaries.
   - The *match automaton* is given as input the sequence of graphemes starting from the *current grapheme*, proceeding in the opposite direction of the edges of $\mathbf{W_G}$, until no further accepting states of the match automata are reachable.
   - Every accepting state which the *match automaton* passes through indicates a possible morpheme boundary (there is always at least one match for the null string, corresponding to a clean split)
4. For each possible morpheme boundary, the right context of the corresponding *sandhi* rule is retrieved, and a depth first search is initiated along the edges of $\mathbf{W_G}$.
   - The depth first search is bounded by morpheme boundaries. Whenever a morpheme boundary or a sink node is encountered, a dictionary lookup is performed.
5. Corresponding to every successful *lexicon* lookup, there exists a split of the word into a morpheme to the right and an unexplored part to the left. $\mathbf{W_G}$ is augmented suitably (see Figure 2(b)) so that the corresponding derivation can be traced by a traversal of $\mathbf{W_G}$ from a source node to a sink node.
   - The *state* of the dictionary determines which parts of it are searchable. The *state* of the dictionary is modified based on the previous state of the dictionary, part of speech, and etymology of the new morpheme.
   - The initial state of the dictionary accommodates searching of all morphemes which can be expected in a word-final position.

6. The *current grapheme* is set to be the first grapheme to the left of the morpheme boundary if a lexicon lookup was successful. Otherwise, it is shifted to the immediately previous grapheme in the word.
7. Repeat steps 3 to 12 until the current grapheme cannot be set as in Step 6 (the source node has been reached). Every possible morpheme starting from the first grapheme of the word is looked up in the *lexicon*. Only those derivations corresponding to valid lookups are correct.

The lexicon lookup is accomplished as follows:

- *basic search word* = Right context of the sandhi in question concatenated with the current depth first search trace.
- If only linear morphology is being handled, only the *basic search word* is searched in the *lexicon*.
- If nonlinear morphology is being handled as well, the *nonlinear affix lexicon* is given as input a sequence of graphemes starting from the last grapheme of the *basic search word*, proceeding from right to left, until no further accepting states are reachable. (Right to left is chosen because only suffixes cause nonlinearity in Bengali – *pratyaya*).
- Every accepting state which the *match automaton* passes through indicates a possible non-linear derivation. The roots of all such possible nonlinear derivations are looked up in the *lexicon*.

The algorithm has a worst case running time and space complexity that is exponential on the number of graphemes (length of the word) for suitably concocted rewrite rule systems. However, Indo-Aryan morphological systems are seldom so productive. Given appropriate morphotactic rules (which is modelled as the way in which the *state* of the dictionary is modified), we may reduce the worst case space and time complexities considerably.

4. Future Work and Conclusion

We are currently working on developing a sufficiently large set of morphotactic rules for the purpose of maximally pruning the *word graph* (minimizing false matches). An empirical assessment of space and time complexity can only be performed once this is complete.

Lookahead for determining if a derivation may at all be valid and further speeding up the morphological analysis may be performed

Finally, computational modelling of derivational morphology, while important in understanding the meaning of unknown words, and grapheme to phoneme mapping in Indo-Aryan languages, is definitely overkill for some applications (eg. spellchecker). In such cases, we may rank all production rules (*sandhi*, *pratyaya* and other affixation rules) using a ranking function. Ranks of inflectional morphological rules are strictly lesser than those of derivational morphological rules, and productivity of rules generally decrease monotonically with rank. We can then set a threshold rank for any application using the morphological analyzer based on which only a subset of the derivation rules (those with rank less than the threshold rank) would be used in morphological analysis.

Acknowledgement

This research is funded in part by Media Lab Asia. The authors are grateful to Prof Jayashree Chakrabarti and Monojit Choudhury for their helpful advice.

References

[1] James Allen, *Natural Language Understanding*, Pearson Education, 2003.

[2] Ronald M. Kaplan and Martin Kay, Regular Model of Phonological Rule Systems, *Computational Linguistics*, 20(3):331-378, September 1994.

[3] Kimmi Koskenniemi, Two-level Model for Morphological Analysis, *Proceedings of IJCAI-83*, pp. 683-685, Karlsruhe, Germany, 1983.

[4] George A. Kiraz, *Computational Nonlinear Morphology*, Cambridge University Press, 2001.

[5] Jonas Kuhn, Compounding and Derivational Morphology in a Finite-State Setting, *Proceedings of ACL 2003*, pp. 192-199.

[6] Claudia Gdaniec et al, Derivational Morphology to the Rescue: How it can help resolve unfound words in MT, *MT Summit VIII, Proc Conf and Workshops, CD compiled by John Hutchins*, 2001.

[7] B. Mayo, *A Computational Model of Derivational Morphology*, Ph.D. Thesis, Univ. of Hamburg, 1999.

[8] Monojit Choudhury and Anupam Basu, A Rule-based Schwa Deletion Algorithm for Hindi, *Proc of Intl Conf On Knowledge-Based Computer Systems*, pp. 343 – 353, Navi Mumbai, 2002

[9] *ITRANS website*, http://www.aczone.com/itrans/

[10] *The CIIL Corpus*, Central Institute of Indian Languages, Mysore
[11] Ł. Dębowski, Zipf's law against the text size: A half-rational model, *Glottometrics*, (4): 49-60, 2002.
[12] G. K. Zipf, *Human Behavior and the Principle of Least Effort*, Addison-Wesley Limited, 1949.
[13] B. Mandelbrot, Structure Formelle des Textes et Communication. *Word*, 10, pp. 1- 27, 1954.
[14] Bamandeb Chakrabarti, *A Complete Textbook on Higher Bengali Grammar*, Akshay Malancha, 1988.
[15] Samit Bhattacharyya et al, Inflectional Morphology Synthesis for Bengali Noun, Pronoun and Verb Systems. *Technical Report IIT/TR/CSE/2003/AB2*, Department of Computer Science & Engineering, IIT Kharagpur, 2003

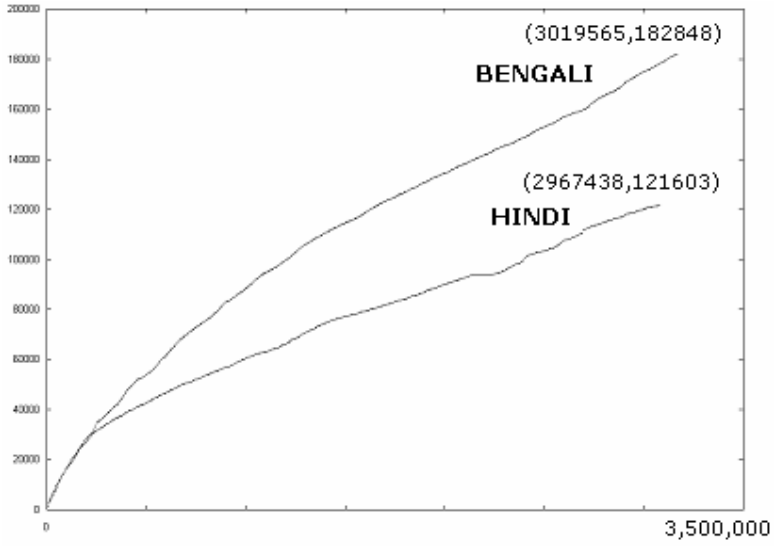


Fig. 1 Corpus size N (X-axis) vs. Vocabulary size V(N) (Y-axis) plot for Bengali and Hindi

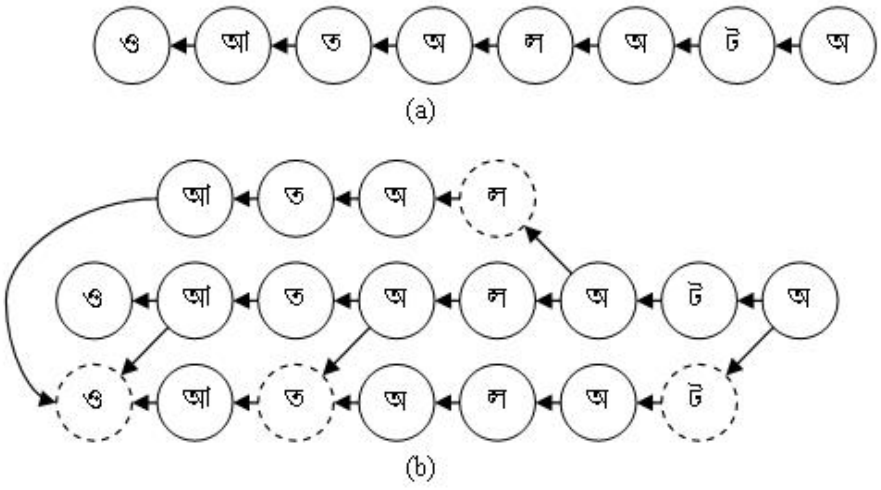


Fig. 2 The Word Graph (a) initially and (b) after morphological analysis. The arrows indicate the direction of writing the graphemes, and not the underlying data structure, which has bidirectional links. The dotted nodes indicate that every incident edge on them are morpheme boundaries.

Generating Nominal Inflectional Morphology in Sanskrit

Girish Nath Jha
Special Centre of Sanskrit Studies
J.N.U. New Delhi - 110 067
girishj@mail.jnu.ac.in

Abstract **-** The present paper presents a generative model for building nominal inflectional morphology (called subanta) in Sanskrit by suitably interpreting and adapting the Pāṇinian sūtras for computing purposes. It also presents a Prolog implementation for such a generator. (The R&D for the work was carried out as M.Phil work completed by the author in J.N.U in 1993 and was supported by the CASTLE project run at SC&SS, J.N.U. under Prof. G.V. Singh)
The resultant model is intended

- as an intermediate system to be input to larger sophisticated NL systems like M(A)T or NLU Systems in the Indian multilingual context, and
- as an end-system for CALT (Computer Aided Language Teaching)

1. INTRODUCTION

The nominal inflectional morphology of Sanskrit has been implemented using PDC-32 Prolog for the DOS environment. In Sanskrit, inflections operate at the level of nominals (subanta) and verbs (tiṅanta). The inflection ending stems are called `pada' (syntactic word). Padas with suP constitute the NPs (subanta-pada), and those with tiṅ can be called constituting the VPs (tiṅanta-pada). In the former, the bases are called `prātipadikas' (PDK) which undergo suP affixations under specifically formulated conditions of case, gender, number, and also the end-characters of the bases to yield nominal syntactic words. The rules for subanta padas are found scattered in Aṣṭādhyāyī mostly in chapters 7-1, 7-2, 7-3, 6-1, 6-4. However, these rules have been treated in the Subanta chapter of Siddhānta Kaumudī (SK) from rule number 177 to 446.

1.1 Previous Work

Mention may be made of the following projects completed under TDIL (Technology Development for Indian Languages) initiative of GOI

- *Desika*: claims to be an NLU system for generation and analysis for plain and accented written Sanskrit texts based on grammar rules of Pāṇini's Aṣṭādhyāyī. It also has a database based on Amarakośa and heuristics based on Nyāya & Mīmāṃsā śāstras. It claims to analyze Vedic (scriptural) texts as well.
- *śābdabodha*: is an interactive application to analyze the semantic and syntactic structure of Sanskrit sentences

The present work needs to be distinguished from the above mentioned projects because it presents a comprehensive solution exclusively for subanta padas only. It takes Pāṇini as the basic formalism and then interpretations from SK and other sources for understanding the intricacies involved in subanta morphology.

2. OPERATION OF NOMINAL INFLECTIONS

Nominal inflectional morphology in Sanskrit consists of nominal bases combining with appropriate case affixes according to number/ gender/ case information and last character of the base. Before we start describing the operations of subanta, a brief note on some of the common technical terms may be in order –

Prātipadika are the crude forms with which the suP affixes combine under specific morphophonemic environments to generate padas or syntactic words. According to Pāṇini any meaningful form of a word which is neither a root nor an affix is called a PDK. They are either primitive (stored in gaṇapāṭha) or are derived through primary (kṛT), secondary (taddhita), feminine (strī) affixations, and also by compounding (samāsa) [1.2.46].

About vacana (number) Pāṇini says that the three affixes in each set like [sU, au, Jas] [am, auṬ, Śas] etc are respectively called singular, dual and plural [1.4.103]. They denote unity, duality and plurality respectively [1.4.21-22]. Some words are only in

plural like `dārāḥ'(wife) and `apaḥ'(water). The numeral `ekaḥ' (one) is in singular only, `dva' in dual only while 'tri' and above are plural only. As in other languages, many words are, by the nature of their use, found to occur only in the singular. The dual is used strictly in the cases where two objects are logically related, whether directly or by the combination of two individuals. When the duality of the objects is well understood (as in the case of pairs of objects), the dual is used (without the word `dva'). For example, `aśvinau'(two aśvins), `akṣiṇī'(pair of eyes), `hastau'(pair of hands) etc.

Vibhakti (case affix) is defined as the sets of three case affixes [1.4.104]. Each case will be marked by its vibhakti for three numbers. The declensional forms show primarily case and number, but they also indicate gender. Though the distinctions of gender are made partly in the stem itself, they also appear in the changes of inflection. Sanskrit has three genders - masculine, feminine and neuter. The only words which show no sign of gender distinction are the personal pronouns of the first or second person, and the numerals above four.

Pada is another crucial term here. Pāṇini [1.4.14] defines it as anything that ends in 21 suP or 9+9 tiṄ affix. The other definition of pada as given in the subsequent three sutras is not relevant here.

In Sanskrit, there are 21 (7x3) case affixes for the purpose of morphological generation of nominal syntactic words (subanta padas). Pāṇini [4.1.2] has listed them -

sU(1-1), au(1-2), Jas(1-3),
Am(2-1), auṬ(2-2), Śas(2-3),
Ṭā(3-1), bhyām(3-2), bhis(3-3),
Ṅe(4-1), bhyām (4-2), bhyas(4-3),
Ṅasi(5-1), bhyām(5-2), bhyas(5-3),
Ṅas(6-1), Os(6-2), ām(6-2),
Ṅi(7-1) Os(7-2), suP(7-3)

The affixes combine with the nominal bases (PDK) (nouns, adjectives, pronouns and numerals) according to the ending characters of the bases. End-character can be a vowel (mātrā) or a consonant (halanta). Each of these classes can have gender specifications from one to three. PDKs are divided into 4 categories - nouns, adjectives, pronouns and numerals.

The correspondence of the first two is so close that they are treated as one category. For these Whitney(1969) [9] provides the following classification stems ending in a, i/u, a/i/u, r, consonants.

1.2 Sample operations: `a' ending masculine nouns

For nom. sing.(1-1),the affix is '-s' (from `sU' [1.3.2]) which in three stages is replaced by visarga (s>r>:). The pada formed would be `rāmaḥ', 'gopālaḥ'etc. The acc. sing.(2-1) termination is `am' which forms `rāmam' after 6.1.107 (ami pūrvaḥ) blocks dīrgha sandhi (vowel lengthening). The rule 7.1.1 provides for instr. sing(3-1) `in' (Ṭā >ā [1.3.7] > in [7.1.2]), abl. sing. (5-1) `āt' (Ṅasi > Ṅas[1.3.2]> as[1.3.8]> āt[7.1.12]), and gen.sing. (6-1) `sya' (Ṅas > as[1.3.8]>sya[7.1.12]). The padas generated are `rāmeṇa', `rāmāt' and `rāmasya' rsepectively for nom. base `rāma'. For dat.sing. (4-1) `ya' is substituted for `e' of `Ṅe' (Ṅe>e[1.3.8]>ya[7.1.3]) which causes the final `a' of the base to be lengthened [7.3.102]. The pada obtained is `rāmāya'. The loc.sing.(7-1) form is `rāme'[6.1.87] i.e. `rāma'+ 'i' (Ṅi>I[1.3.8]) => `rāme'. Voc.sing.(8-1) form is `he rāma' (he rāma + sU[2.3.49] => he rāma + 0[6.1.69]).

The Nom-dual (1-2) form is rāma + rāma = rāmau (rāma + au ->rāmau[6.1.88]). The Pāṇinian injunction here is that if there are more than one word of the same form and vibhakti then only the last one is retained [1.2.64]. Similarly, acc-pl (2-2) form is rāma + au -> rāmau[6.1.87] auT->au [1.3.7]). The 3-2,4-2,5-2 termination `bhyām' yields the pada rāmābhyām[7.3.102]. Gen. and loc. dual (6-2,7-2) affix os -> oḥ yields rāmeyoḥ[6.1.77] (rāme[7.3.104] + oḥ). The 8-2 form is like nominative with the prefix `he'.

For nom-pl (1-3), the termination `as' (from `jas') by dīrgha gives rāmās or rāmāh. Acc-pl (2-3) `as' (from`Śas'[1.3.8]) combines with the base: rāma + as ->rāma + an (as->an[6.1.103]) -> rāmān[6.1.102]. Instr-pl (3-3) affix `bhis' is substituted by `ais'[7.1.9] to give rāmaiḥ[6.1.88]. The 4-3,5-3 affix `bhyas' combines with a modified base (`a' > `e'[7.3.103]). Thus rāma + bhyas -> rāme + bhyaḥ -> rāmebhyaḥ. For 6-3, affix `am' gets 'nUṬ' augment with the final `a' of the base getting lengthened - rāma + am -> rāmā[6.4.3] + nām[7.1.5] (n->ṇ[8.4.2]). Before loc-pl (7-3) vibhakti `su'(suP->su[1.3.3]), the final `a' of the base is changed to `e'[7.3.103]. Thus rāma +suP -> rāme + su -> rāmeṣu (s->ṣ[8.3.57]). Voc-pl (8-3) is like nom-pl(1-3). The form is 'he rāmāḥ'.

3. PROLOG IMPLEMENTATION

Implementation of Pāṇinian formalism as explained in SK was done using Prolog (PDC-3.2). For each

input PDK the program generates 24(21+3) śabdarūpa and also parsed constituents with details.

3.1 Procedural Details

On running the `nomor.exe', the program will ask for an input, a nominal base (PDK), and flash a set of Roman symbols at the right-hand corner of the screen to handle the input-output mechanism in Roman as well (in case, facilities for devanagari does not exist). On pressing `Enter', the script program will first convert the input into its Nāgari equivalent and search in a program specific database to return other required information like the category/gender. In case of a nominal base being used in all the three genders, the program, with the help of a small menu will ask the user to enter the gender for which he/she would like to see the subanta paradigm. In case, the input nominal base is not found in the database of PDKs, the program will query about the category and the gender/s associated with the input through small menus.

The information pertinent to the PDK (either returned from the database or sought from the user), will be fed into the morphological analyzer and thence to the lexicon of case affixes. These two components have the grammar implementation. The morphological analyzer will, first of all, extract the final character/string of the nominal base with the help of user-defined predicates called `last_str', and `word_tail' which use standard predicates `frontstr' and `str_len'.

The terminating string can be either a vowel represented by its mātrā, a consonant represented by its marker called `halanta', a consonant cluster, a conveniently isolated part of the input string or the whole string itself. The clause section of the program has specifications for vowel - mātrā, consonants, and other specific constituents as final strings of the nominal base. The program will search for the terminating strings/whole words among the clauses section and return the change/s (if any) in the nominal base before affixation. In case, program does not find any clause for the said final string/ whole word, the last clause (that is the general rule) will bound the variables with values.

In the next stage, the program will go to the lexicon of case affixes and look for a possible match among various clauses. For each category of the terminating string/character, there would be a set of case affixes with their appropriate reduced forms, descriptive details and rules etc. Otherwise, a general set of affixes and other values will return the outputs.

Next, the Sandhi rules are applied to arrive at the macro forms. Here, the modified (modifications if any) stem and the reduced stem (reductions if any) pass through morphemic processing under specific rule governed environments to yield the final output called `pada' (syntactic word/finished word). This output will once again pass through the 'Script' program for re-conversion into its original fold. This operation will be repeated for a maximum of 24 (21+3 case conditions) forms and display *subanta* forms in a paradigm arranged in the case-number order. Parsing details can be obtained by selecting a form.

All these components,i.e, the script, morphological analyzer, case_aff_lexicon and the sandhi component operate within a super component called the `sound_class' which has basic Pāṇinian pratyāhāras or sigla denoting important sound classes being used in various components.

3.2. Main Program

The main program consists of three modules –

- `call.pro': contains main predicates which make windows, menus etc, read the inputs and associated information (if any), call predicates from other files and display the generated word forms along with parsed constituents on the screen.
- `morph.pro': examines the end character of the input string. Based on its type (various kinds of vowels and consonants), it may effect modifications in the input string (PDK). The output from this file would become the first input (W1) in the morphophonemic component.
- `c_aff_lex.pro': lexicon of nominal case affixes based on the various types of end characters of a nominal base. It assigns sets of values of case affixes to the modified PDK according to number, gender and case.

Outputs from this component will go as the second input (W2) into the sandhi component.

3.3 Accessory Programs

The core components (morphological analyzer and the case_aff_lexicon) make calls to the following accessory components –

- script.pro : for Roman-Devanāgarī inter-change
- sound_cl.pro : the sound class program containing Pāṇinian Pratyāhāras
- nom_lex.pro : program-specific database of Nom_Bases
- mph_comb.pro : sandhi rules

3.4 Technical/non-technical problems

The following problems were encountered –

- Interpreting Pāṇini: clarifying some of the ambiguous explanations from SK by cross-referencing Katre's edition of Aṣṭādhyāyī, Kale's grammar and a few school level Sanskrit grammar books.
- morphophonemic combinations : multiple operations for combining elements at different stage was expensive speed-wise.
- GIST compatibility: providing a facility for the execution of the program on PCs without the GIST card facility was particularly problematic. A separate program had to be written for both sandhi as well as for the script handling.

3.5 Limitations

The program may give incorrect results for some rarely used PDKs in commonplace Sanskrit. The parsed constituents and procedural details try to follow the Pāṇinian methodology as far as practicable. These details may not be of much use and interest to a child learner, but certainly so for an advanced learner who has some background in the Pāṇinian system.

The program allows input - output mechanism in the scripts of major Indian languages or Roman. The input must not be a mix of scripts. Secondly, the names and concepts etc. to be input as nominal base should be peculiar to Sanskrit language or other Indian languages which are culturally on the same plane. Other inputs (if required) which are to be selected from menus should also be correct otherwise the outputs may not be according to the Pāṇinian formalism.

4. CONCLUSION

The paper thus attempts to present a computational system for nominal inflectional morphology of Sanskrit by applying the Prolog based NL techniques. Though there are limitations on various counts as mentioned above, yet the resultant program-model can be used in machine aided language pedagogy and for NLP research purposes. It can also be suitably modified and adapted as a system intermediate to larger and more sophisticated NL systems for Indian languages besides Sanskrit.

REFERENCES

[1] Cardona, George, "On translating and formalizing Pāṇinian rules" (*Journal of Oriental Institute*, Baroda, vol 14, 1965, 306-14)

[2] Cardona, George, "Pāṇini: His work and its traditions ", Motilal Banarasidass, Delhi, 1988)

[3] Deshpande, Madhav M. "Pāṇini in the context of Modernity" (*Language and text* ed. R.N.Srivastava et al, Kalinga Publications, Delhi, 1992)

[4] Gal, A., Lapalme, G., Saint-Dizier, P. & Somers, H., "Prolog for Natural Language Processing" (John Wily and Sons Ltd., West Sussex, England, 1991)

[5] Grishman, R., "Computational Linguistics: An Introduction" (*Studies in NLP*, sponsored by ACL, Cambridge Univ. Press, New York Univ. 1986)

[6] Kale, M.R., "A Higher Sanskrit Grammar" (Motilal Banarasidass, 1987)

[7] Kapoor, Kapil, "Pāṇini Vyakarana: Nature, Applicability and Organization" (course notes for NLP-91, IIT-Kanpur, 1991)

[8] Katre, Sumitra, M., "Aṣṭādhyāyī of Pāṇini" (first Indian edn. Motilal Banarasidass, 1989)

[9] Whitney,W.D. "Sanskrit Grammar" (Motilal Banarasidas, Delhi, 1969)

A Two-level Morphological Processing of Oriya

Kalyanamalini Sahoo
ILTS Lab,Indian Institute of Science, Bangalore
kalyani@mgmt.iisc.ernet.in

Abstract **-** Morphological analysis dealing with analysis and generation of different types of word forms, plays an important role in applications such as spell checking, electronic dictionary interfacing and information retrieving systems. This paper discusses a two-level morphological processing of Oriya. It proposes a model for designing a morphological analyzer for Oriya, which can provide lexical, morphological and syntactic information for each lexical unit in the analyzed word form.

1. INTRODUCTION

Analysis and generation of word forms is a crucial and basic tool in the processing of natural languages. Morphological analysis dealing with analysis and generation of different types of word forms, plays an important role in applications such as spell checking, electronic dictionary interfacing and information retrieving systems. In this context, some applications, like spelling correction, do not need more than segmentation of each word into its different component morphemes along with their morphological information. However, there are other applications such as lemmatization, tagging, phrase recognition, and determination of clause boundaries, which need an additional *morphosyntactic parsing* of the whole word. Morphological analysis of words is indispensable when dealing with agglutinative languages like Oriya. This paper discusses a two-level morphological processing of Oriya. It proposes a model for designing a morphological analyzer for Oriya, which can provide lexical, morphological and syntactic information for each lexical unit in the analyzed word form. It draws out a finite-state machine that accepts valid sequences of morphemes in a word and rejects invalid ones. Such identification of sequences has a number of practical applications like spell checker, machine translation etc. In this paper, we consider the processing of Oriya nominal forms only. We can use the Finite State machines to solve the problem of morphological recognition, determining whether an input string of morphemes makes up a legitimate Oriya word or not. We do this by taking the morphotactic Finite State Automaton (FSA)s and plugging in each word into the FSA. We do this via two-level morphology (TLM). Two-level morphology is based on three ideas [1]:

- Rules are symbol-to-symbol constraints that are applied in parallel.
- The constraints can refer to the lexical context, to the surface context, or to both contexts at the same time.
- Lexical lookup and morphological analysis are performed in tandem.

TLM represents a word as a correspondence between a lexical level, which represents a simple concatenation of morphemes making up a word, and a surface level, which represents the actual spelling of the final word. Morphological parsing is implemented by building mapping rules that map morpheme sequences like *pilA-mAne* 'children' on the surface level into morpheme and feature sequences like 'child + plural', at the lexical level. The automaton used for this mapping is the finite-state transducer or FST. For TLM we view an FST as having two tapes; the upper or lexical tape contains the input symbols and the lower or surface tape contains the output symbols. FST defines a relation between sets of symbols and maps them via a finite state automaton.

Our machine provides a basic level of processing extended by language specific morphological knowledge sources. There are two basic components to the system: the engine itself, and the knowledge sources (i.e. lexicons, rule files and grammar files). This paper focuses on the latter component – the morphological knowledge sources. The engine is capable of two basic modes of operation: *recognition*, in which a fully inflected word is processed by the system to arrive at a description of the word's morphological

decomposition(s); and *generation*, in which a specification of underlying morphemes is processed by the system to produce the corresponding surface forms of the word.

The remainder of this paper is organized as follows. Section 2 discusses the morphological structure of Oriya nominal forms. Section 3 describes the architecture for morphological processing, specifies the phenomena covered by the morphological analyzer, explains its design criteria, and presents the processing details. Finally, the paper ends with some concluding remarks.

2. ORIYA NOMINAL FORMS

Oriya is a syntactically head-final and morphologically agglutinative language. A number of morphemes carrying different grammatical functions get affixed to the nominal root to make a nominal form. The major affixing categories found in a nominal form are: numeral (Nmrl), classifier(Cl/Clas), quantifier (Quan), number marker (Nmb), negation marker (Neg), qualitative affirmative marker (Q.Aff), Case marker and postpositions (PP)etc. E.g.

(1) pilA-dui-TA-jAka-nku
child-two-Cl-Quan-Case
'To those two children'
(2) apa-karma-mAna
NEG-deed-pl. [Pl=plural marker]
'Bad deeds.'

2.1. Pronouns

In Oriya all the pronouns can be used with reference to nouns of all genders. Like nouns, the pronouns can have Case features too. Pronouns are marked for person, number, honorificity and animacy. They can be [±definite]. For all pronouns, inflections are usually suffixed to the root. Reduplicated pronominal forms like *kichhi kichhi* /something something/ 'certain amount' are also used in Oriya.

2.2 Number, Numeral and Quantifier

Oriya distinguishes between the number marker and numerals. Number is realized either as a numeral or as a nominal inflection, but not as both; e.g.

(3) a. pAncha-(TA)- bhAi b. bhAi -mAne
five Cl. brother brother Pl.
'Five brothers.' 'Brothers.'
c.*pAncha-(TA) bhAi -mAne
five Cl. brother Pl.
'Five brothers.'

Oriya has singular, dual and plural numbers. Usually, singularity is indicated by the bare stem, by the numeral *eka,* by the [classifier+numeral] sequence like *goTA-e, goTi-e, goT-e,* by the singular suffix *-e, -ka*, etc. [2]. E.g.

(4) a. bastA-e chAuLa b. chauLa bastA-ka
sack-one rice rice sack-one[+def]
'A sack of rice.' 'The sack of rice.'

Duality is marked by the suffix *duhen* 'both'. E.g. *Ame-duhen* / we both/ 'Both of us.'

Plurality is marked by the suffix *-e* in the nominal, e.g. *lok<u>e</u>* 'people', the suffix *mAne* or *mAna*, e.g. *pilA-<u>mAne</u>* 'children', prefixing or suffixing *sabu* 'all', *samasta* 'all', e.g. *<u>sabu</u>-pilA* /all child/ 'all the children', *Ame-<u>sabu</u>* 'we all', the morphemes denoting entirety, multitude, or quantifying amount such as *-jAka*, *-taka* (in the case of uncountable nouns) and *-guDA*. E.g. *khira-<u>taka</u>* 'the whole amount of milk', *masA-<u>guDA</u>* 'the mosquitoes'.

Numerals occur in a position immediately preceding or following the root noun, while number markers can occur immediately following the nominal root as well as at the final position of the nominal stem following the classifier. Number markers and quantifiers are in complementary distribution, hence, occupy the same positional slot in the nominal form.

2.3 Case

Like other Indo-Aryan languages, Oriya follows the Case system of Sanskrit. The Cases are named and defined after the Sanskrit grammar. There are eight Cases in Oriya: Nominative, Accusative, Instrumental, Dative, Ablative, Genitive, Locative, and Vocative. Nouns and pronouns are marked for Case indicating some grammatical function. However, pronouns lack the vocative Case. The Case morphemes are realized as follows in Table 1.

2.4 Postpositional words

Postpositional words like *pare* 'after', *kari* 'by', *nimite* 'for, *parjyante* 'upto', *pAin* 'for', *prati* 'to', 'against', *byatita* 'without', *binA* 'without', *boli* - literally speaking bhitare 'in', 'inside', lAgi 'for',

sahite 'with', etc. occur at the final position of the nominal stem. Case markers and postpositional words are mutually exclusive and occur in the same positional slot in a nominal form.

Table 1. The Case morphemes in Oriya

Case	Singular	Plural/[+Hon] sg
Nominative	-	-e
Accusative	*ku*	*nku, mAnanku*
Instrumental	*re, dwArA, dei*	*re, dwArA, dei*
Dative	*ku*	*nku, mAnanku*
Ablative	*ru, ThAru*	*mAnankaThAru*
Genitive	*ra*	*nkara,mAnankara*
Locative	*re, ThAre*	*mAnankaThAre*
Vocative	*he, bho*	-

2.5 Qual Aff markers & Qual NEG markers

Qualitative affirmative markers like *su-, sat-, sad-* and qualitative NEG markers like *apa-, ku-, duh-, a-, bad-* are used. They are usually prefixed to the noun root. E.g. *su-guNa* 'good quality', *sad-byabahAra* 'good behaviour', *a-sundara* 'ugly' *bad-abhyAsa* 'bad habit', *apa-karma* 'bad deed'. Affirmative markers and NEG markers are mutually exclusive. Only one item can be attached to the noun Root at a time.

Summarizing, the occurrence of items in An inflected nominal form can be shown as follows:

(5) (NEG)/(Qual Aff) – Root N – (Numeral) – (Clas) – (Quan)/(Number) – (Case)/(PP).

2.6. De-verbal nouns

De-verbal nouns are formed by the affixation of a nominal suffix to the verbal root. E.g. *randh-A* 'cooking', *AN-ibA* 'bringing' *chaDh-ALi* 'climber', etc.

3. FST FOR MORPHOLOGICAL PROCESSING

As we discussed above, there is a rich structure in these morphological sequences, and in this paper we will model it by using a deterministic finite-state transducer (FST). Such a morphological analyzer has to consider three main aspects, as discussed by [3]Ritchie *et al.*(1992), [4]: (6)

i) Morphographemics or morpho-phonology:

This term covers orthographic variations that occur when linking morphemes.

ii) Morphotactics: Specification of which morphemes can or cannot combine with each other to form valid words.

iii) Feature-combination: Specification of how these morphemes can be grouped and how their morphosyntactic features can be combined.

As a consequence of the rich morphology of Oriya, we try to control most of the morphotactic phenomena in the morphological segmentation phase. The morphological analyzer created by (Ritchie et al*.,* 1992), although does not adopt finite state mechanisms to control morphotactic phenomena, their two-level implementation incorporates a straightforward morphotactics, reducing the number of sublexicons to the indispensable (prefixes, lemmas and suffixes). But such an analysis will not be sufficient to account for an agglutinative language like Oriya, as it would create many unacceptable forms. Therefore, we separate sequential morphotactics (i.e., which sequences of morphemes can or cannot combine with each other to form valid words), which will be recognized by means of continuation classes, and non-sequential morphotactics like *long-distance dependencies* that will be controlled by the word-grammar. Since we cannot list every word in the language, computational lexicons are structured as a list of stems and affixes with a representation of the morphotactics. One way to model morphotactics is the finite-state machineries. We use a deterministic FST to solve the problem of morphological recognition. Two-level rules can be applied for both recognition and generation of surface forms. The morphological analysis will map each inflected word into a unique canonical form (the root form of a Noun) plus a list of morphological features such as person, number, etc.

The two level model [5] views each word in the language as having two simultaneous representations (or correspondences): the lexical and the surface. The former is characterized as an underlying concatenation of morphemes in their neutral forms (generally corresponding to the lemma). The latter (surface) form represents the actual orthographic form of the word in the language. Rules, specified in a fairly transparent format, mediate the L:S pairs. Finite state tables corresponding to these two-level rules are processed by an algorithm which simulates composition of rule automata with the lexicon, transducing the L:S word forms in both directions.

Oriya grammar is composed of a set of rules. A set of definitions is compiled as a finite-state transducer where the lower part of the transduction (left projection) is a string and the higher part (right projection) is also a string. In such a finite-state machine, an edge carries strings of input and output. From an analysis point of view, the inflected form is decomposed into lexemes and morphemes by traversing the network using the left projection, and concatenates the symbol of the right projection to build the output. From a generation point of view, the network is traversed based on the right projection symbol equality, and concatenating the string elements of the left projection to build the inflected word form. Formally, the Oriya transducer T is a tuple (I, O, S, s, F, δ) where I is the input alphabet, O the output alphabet, S is a finite set of states, s is the initial state, F is a set of final states, δ is the transition function from $S \times I \times O$ to S. The input alphabet I and the output alphabet O are (finite) set of characters.

For TLM, we visualize an FST as a two-tape automaton which recognizes or generates pairs of strings. FST maps sets of symbols via a finite state automaton. FST defines a relation between sets of strings; it reads one string and generates another. The symbols of the FST are *pairs* of symbols, one for each of two "tapes" or levels. The upper or lexical tape is composed from characters from the left side of the a : b pairs, the lower or surface tape is composed of characters from the right side of the a : b pairs.

An FST accepts a language over pairs of symbols, as in the following:

Table 2. mapping over pairs of symbols

Stem	N	Pl.
PilA	Σ	*{mAne, guDA, jAka, taka,..}*

The FST has the following components:
Recognizer: decides if a given pair of representations fits together "OK". **Generator:** generates pairs of representations that fit together. **Translator:** takes a representation on one level and produces the appropriate representation on the other level. **Dictionary, text:** each consist of a sequence of items – items of the dictionary are expressed according to an alphabet which consists of {*ka…kshya*}, 0 (empty character), + morpheme boundary character, set of grammatical features e.g. pl. for {*mAne, guDA*}. –items of text are expressed by a subset of this alphabet { *ka…kshya*, 0}

The arcs should be labeled as the following:

Qual Aff: {*su-, sad-, sat-*}

NEG: {*apa-, ku-, duh-, a-, bad-*}
Nmrl: {*dui, tini, chAri,..*}
Nmb: {*-e, -ka, duhen, mAne, guDA,..*}
Clas: {*TA*}
Case: {*ku, re, ra, ru, ThAru, ThAre*, …}
PP: {*pare, kari, nimite, prati, byatita*, …}
N suf: {*A, ibA, ALi*}

4. CONCLUSION

An efficient finite-state morphological analyzer has been described. We specify the co-occurrence restrictions of the morphemes in nominal forms and use the FST to solve the problem of morphological recognition and generation; determining whether an input string of morphemes makes up a legitimate Oriya word or not. Such identification of sequences have a number of practical applications like spell checker, machine translation, etc. We think that our design could be interesting for the treatment of other agglutinative languages too.

REFERENCES

[1] Kartunnen, L & K. Beesely, "A Short History of Two-Level Morphology", Paper presented at the ESSLLI 2001, Helsinki, 2001.

[2] Sahoo, K, "The DP-Analysis of English and Oriya Noun Phrases". M.Phil dissertation. CIEFL, Hyderabad, 1996.

[3] Ritchie, G., S. Pulman, A.Black, G.Russel, *Computational Morphology: Practical Mechanisms for the English Lexicon*. ACL-MIT Series on Natural Language Processing, MIT Press, 1992.

[4] Sproat, R., *Morphology and Computation*. ACL-MIT Press series in Natural Language Processing, 1992.

[5] Koskenniemi, K, *Two-level morphology: A general computational model for word-form recognition and production.* Publication 11, University of Helsinki, 1983.

Parsing and generation of verbal nouns and other verb-derivatives in OIA

Saranya Saha
Department of Information Technology
Jadavpur University
Calcutta 700032 INDIA
email: saranya_saha@hotmail.com

1. INTRODUCTION

In Indian languages, inflectional morphology of nouns can strictly govern semantics, and pragmatics is involved in both noun inflections as well as verb inflections. When we come to the process of derivation, the factors involved are more. The numerous nuances and subtleties in meaning with which the process of forming derived words are related make the situation very complex to analyse. However, a proper analysis can always bring rewards. Here we shall mainly deal with the morphology of a class of derived nominal stems and some other derivatives of verbal roots (except derived roots) in Sanskrit, which is an Old Indo Aryan language and traditionally considered as the predecessor of all the Modern North Indian languages. These nominal stems in Sanskrit are derived from verbal roots through the addition of some noun-forming suffixes and declined just like any other nominal stems.

When in a language, nouns are formed from verbs denoting certain acts in various senses like the action, the agent, the instrument, the object, the location, etc., and standard suffixes are applied after each verbal root in fixed senses, then it is economical to parse these words into the verbal roots and the standard suffixes and search for the roots in the lexicon, instead of storing each such verbal nouns along with the verbs.

When the suffixes are non-standard, are not applied after all the verbal suffixes and the meanings are not simple, then the parsing process as well as the generation rules can become quite complex. In these cases, preferably the nominal stems must be stored in the dictionaries and treated as indivisible. Often, the phonological laws and morphological patterns are not fully explained with such a process. However, even in Rule Based Language Processing, it makes sense to use approximate explanations and speed up the process of analysis (parsing) and synthesis (generation). As an overhead, in the dictionary, we need to store these entries along with their meanings. But, the number of such cases where the general rules would not apply is small and hence the size of the lexicon is not increased significantly.

It is difficult however to select a set of suffixes as 'general' and mark the rest as 'exceptional' or 'special', as the real scenario is much more complex than being a "yes no" (general/not general) case. Here we shall look at different suffixes, which are fairly general and then consider the special cases. Some of the exceptional words are given as it is, while in others the suffixes are given and as such we need to apply those suffixes to obtain the words. We shall follow the system of Pāṇini to extract the general suffixes and exceptional words. In this system, the bases (prakṛti) and the suffixes (pratyaya) are listed along with their various meanings and then the exceptional words are given.

In Sanskrit, many different types of nouns are derived from a verbal root using different types of suffixes. These verbal suffixes are generally known as "kṛt" suffixes. The simplest process in forming a word using verbal suffix is to add the suffix to the verbal root, the word being thus formed, without any other operation taking place. In some cases, the verbal root needs to undergo some change before it can be joined to the suffix. In others, an augment is placed in between the verbal root and the suffix. Sometimes, certain portions of the verbs are elided. Sometimes the verbal root would be appended to another word, when the action has some connection with the entity or object that word is describing. This word may be added before the verbal root, after which the suffix would be joined, or there may be some augment in between the word and the verbal root. The same suffix may be added to a verbal root following an attendant word without an augment in one case, and to another root following some other attendant word, with an augment in some other case.

2. THE GENERAL RULES

The suffixes that come after all verbal roots, the senses in which these are used, and the

augmentations and modifications that they trigger are given below.

1) Action to be performed necessarily or that which is fit to be done or appropriate to do:
anīya: used after: all roots, augments: none, modifications: simple vowels changed to guṇa letters
tavya: used after: all roots, augments: 'i' augment for a class of roots, modifications: simple vowels changed to guṇa letters
ya: used after: all roots ending in a vowel except 'ṛ' or ending in a bilabial preceded by 'a', augments: no augment, modifications: 'ā' changed to 'ī', simple vowels changed to guṇa letters
ya: used after: all roots ending in consonants (other than bilabials preceded by 'a') and 'ṛ', augments: no augment, difications: simple vowels changed to vṛddhi letters
2) Agent:
aka: used after: all roots , augments: no augment, modifications: simple vowels changed to vṛddhi letters
tṛ: used after: all roots, augments: 'i' augment for a class of roots, modifications: simple vowels changed to guṇa letters
3) Past Participle:
ta: used after: all roots, augments: 'i' augment for a class of roots, modifications: no modification
tavat: used after: all roots, augments: 'i' augment for a class of roots, modifications: no modification
4) Present Participle:
at: used after: all roots, augments: the 'vikaraṇa' augment, modifications: guṇa of root final simple vowel
āna:
used after: all roots, augments: the 'vikaraṇa' augment and 'm' augment for stems ending in a, modifications: guṇa of root final simple vowel
5) Habitual Agent:
tṛ: used after: all roots, augments: 'i' augment for a class of roots, modifications: simple vowels changed to guṇa letters
6) Purpose/ Future Action (Infinitive):
tum: used after: all roots, augments: 'i' augment for a class of roots, modifications: simple vowels changed to guṇa letters
7) Action:
a: used after: all roots, augments: no augment, modifications: simple vowels changed to vṛddhi letters, palatals changed to gutturals
a: used after: verbs ending in simple vowels expect 'a',
augments: no augment, modifications: simple vowels changed to guṇa letters
ti: used after: all roots (the word formed is in feminine gender), augments: no augment, modification: no change
8) Instrument/Locus:
ana: used after: all roots, augments: no augment, modifications: simple vowels changed to guṇa letters
a: used after: all roots, augments: no augment, modifications: simple vowels changed to vṛddhi letters, palatals changed to gutturals
9) Past act:
tvā: used after: all roots, augments: 'i' augment for a class of roots, modifications: no change
am: used after: all roots (in case of doubling), augments: no augment, modifications: simple vowels changed to vṛddhi letters.

The word'guṇa' refers to the change of sounds 'i' / 'ī', 'u' / 'ū' and 'ṛ' to 'e', 'o' and 'ar'. The word 'vṛddhi' refers to the change to sounds 'ai', 'au' and 'ār'.

The class of verb roots for which an 'i' augment comes in case of suffixes beginning with a devoiced consonant[1] are: all roots except monosyllabic roots ending in a vowel[2] and some other exceptional roots[3]. The 'vikaraṇa' augment is 'a', 'ya', 'nu', 'na', 'u' or 'nā' depending on the class of the verb root. The sandhi rules must be applied in the generation process as well in analysis. The general rules of sandhi are the following:
1) Two homogeneous vowels are replaced by a single long vowel homogeneous to each.

2) A corresponding guṇa letter is the single substitute of 'a' followed by a simple vowel.
3) A corresponding vṛddhi letter is the single substitute of 'a' followed by a diphthong.
4) The 'a' of a prefix is dropped when 'e' or 'o' of a root follows.
5) A long or short 'i', 'u', 'ṛ' and 'ḷ' is changed to the corresponding semivowel before a vowel.
6) The diphthongs- 'e', 'o', 'ai', 'au' are converted to 'ay', 'av', 'āy' and 'āv' respectively.
7) A dental sound changes to a palatal letter or a cerebral letter when in contact with a palatal letter or a cerebral letter respectively.
8) A mute, when followed by a voiced mute is changed to the corresponding unaspirated voiced mute.

[1] except ten special suffixes
[2] other than 'ṛ' or 'ū' (excluding twelve roots)
[3] 102 roots ending in consonants

9) A mute, when followed by a devoiced mute is changed to the corresponding unaspirated devoiced mute.
After a surd letter, ‘s’ is changed to ‘ch’.
10) A nasal is changed to the one agreeing with the following non nasal mute.

The general rules for cerebralisation are as follows:
1) After a ‘i’ (short or long) or ‘u’ (short or long), a dental ‘s’ is changed to cerebral ‘ṣ’.
2) The same change occurs when the source and the dental letter is immediately separated by a ‘n’ augment.
3) After a ‘ṛ’, ‘r’ or ‘ṣ’, the dental letter ‘n’ is changed to the cerebral letter ‘ṇ’.
4) The same change occurs in one word when there is an intervening vowel or ‘h’ / ‘y’ / ‘r’ / a guttural sound / a bilabial sound or the prefix ‘ā’ or the augment ‘n’.
5) The changes described in 3 and 4 occur even when the source is in one member word, the conductor (the allowable intervening element) is in another member word and the dental letter in a third member of a compound word.

3. THE COMPUTATIONAL MODEL

The morphological analyzer and the morphological generator are two separate modules, which may be integrated within a larger processing system for the language on the whole. The morphological analyser would take as input a word in the form in which it occurs in a sentence. When a dictionary search for the word as such would fail, then it would be analysed. If as input, we can get the part of speech of the word, then we know what type of analysis is expected. In the absence of such information, one part of speech would be assumed and analysis done with the other, then a second part of speech is assumed, till we get the desired result. In this case we would be concerned with derivatives of simple or complex verbal roots.

If the derivative is a noun, then it would be present in an inflected form. The declensional suffix and the nominal stem would be isolated, and then the stem would be searched in the lexicon. If the search fails, then the stem is known to be derived either from some verbal root or it is a secondary nominal stem derived from a primary noun. If the module for determining the primary noun from a secondary noun fails, then the word is a primary noun and our module would determine the verb root of that derived word. In case the stem is a secondary noun, then the source primary noun may be indivisible or may be derived from a root. In the last case, our module would determine the verbal root and the sense of the suffix. Therefore the analysis module would take the nominal stem derived from a verb root as input and give as output the verb root and the sense of the suffix. If the derivative word is not a noun, then it would be fed straight to this submodule. If the word is irregularly formed or formed through the addition of a special suffix, then the search in the dictionary which is done earlier would succeed and as such this module would not be invoked. This module is invoked only when the verb root and the suffix generate the word using some general rule.

In case of generation, a lexicon search is necessary, before attempting to generate a word, to check whether the word is an irregularly formed word or formed using an exceptional rule. If the search yields positive results, no attempt would be made to generate the word. The lexicon must then contain the irregular word as an auxiliary entry under the root and the sense of the suffix must be mentioned there. The output of our generation submodule is a primary nominal stem or some other verbal derivative, which is either fed to a module to generate the secondary stem, or is inflected using the declensional suffix (or fed in to form a compound) or is fed to the next subprocessor.

The design of the lexicon would be as follows. For analysis, we would have a lexicon of verbal roots and other indivisible words and a list of suffixes, and the input would be matched to a root-suffix pair. The lexicon must have entries for the irregular derivatives of the root mentioning the sense of the suffix or the derived word under each root entry. The input to the generation module would be a root and the sense of the suffix. The proper suffix is selected from a list of suffixes and then applied.

This design is independent of the way the lexicons are physically designed. The model is a high level abstraction and the real complexity of the algorithm would depend on the lower level design and implementation of this model. We are bothered more about accuracy of the output in this high level design.

4. THE ALGORITHM

A. Generation

This module is relatively simple. The input would be a verb root (either simple or derived), an optional set of prefixes in order, and the sense in

which the suffix needs to be used. That sense would be one of the categories of the nineteen suffixes listed above. From the category, any suffix which apply to this root is chosen. Then the augment is determined from the chart. Modifications are made, one following the other. Finally, the prefixes in order, the modified root, augment and the suffix are added using the rules of sandhi. At each conjunction point, the pair of letters are considered, and the proper sandhi rule is chosen based on the pair. (Only one of the rules in the list applies solely to prefix root sandhi.) Finally, the dental 's' is checked for cerebralisation and after the necessary change, the dental 'n' is checked for cerebralisation.

B. Analysis

The suffixes used here are 'tavya', 'tavat', 'anīya', 'tum', 'tvā', 'āna', 'aka', 'ana', 'ti', 'ta', 'tṛ', 'ya', 'at', 'am' and 'a' of several types. When the analysis submodule gets an input, the suffix is identified first. The suffixes may be tried in the order they have been listed in this subsection. If no full syllable is left, a smaller suffix is tried. The dental nasal 'n' of the suffix may alternate with the cerebral nasal 'ṇ'. In the latter case, a leftward search for a cause is done till the cause letter, or a non-allowable intervening letter or the beginning of the word is reached. If the source is found, then the suffix is accepted, otherwise it is rejected.

After extracting the suffix, prefix(es) which may be possibly present are extracted. Roots (without an augment) after modification are either monosyllabic or disyllabic (in case of the class of reduplicated roots). If the portion remaining after extracting the suffix has more than one syllable, then a search for the prefix may be made. Before extracting a prefix containing a cause of cerebralisation, a check for a cerebral nasal 'ṇ' is done rightwards, till a non-allowable intervening letter or the end letter or the 'ṇ' is reached. If the affected letter is found, it is changed back to 'n'. If after extracting a prefix, the remaining portion has more than one syllable, then the process of extracting a prefix is repeated.

The remaining portion may be the modified and augmented form of a candidate root. Depending on the suffix, the augment may be extracted and the demodification process applied. If the remaining portion is at least a syllable in length, then it is a candidate root. It is searched in the lexicon and if it is found, then the analysis is correct. Otherwise, we backtrack to the immediately preceding step of choosing a prefix, and change the option, and so on till we get the real root. If all the iterations fail, then the result is a 'no'.

In case of extraction of prefix or augment or in case of demodification, sandhi rules must be taken care of. The reverse of the sandhi rules would be taken into consideration.

5. COMPLEXITY

The generation process runs in constant time, since no search is involved there. For analysis, the maximum number of times a search in the lexicon may be done is a constant. Hence the time taken for analysis is of the order of the time taken for a search. The dictionary to be searched would contain all the simple verb roots as well as the derived verb roots like desideratives, causatives, intensives, denominatives, etc. The lexicon would still be limited to a few thousand words, and so, even if the search is linear, the time taken is not appreciable. For a binary search, the time is further reduced.

6. ADVANTAGE

The total number of suffixes used in the so called "Classical Sanskrit" language is around one hundred. Most of these suffixes, however, are used after very few selected roots. We have shortlisted about nineteen suffixes from this list. The nineteen general suffixes we have chosen are applied after almost all the verb roots. There are single prefixes and combination of prefixes which are applied after roots. On an average, about ten different prefixes and their combinations are applied after each root. The total number of roots (which are divided into ten classes) is more than one thousand. The total number of words which are formed on applying these nineteen suffixes after these thousand roots preceded by valid combinations of prefixes is of the order of one hundred thousands (100000), and more than two thirds of the total number of words that are formed using the verbal suffixes. This is the major advantage of the system.

REFERENCES

[1] Böhtlingk, O., Panini's Grammatik, Motilal Banarsidass, 1998

[2] Devasthali, G.V., The Anubandhas of Panini, University of Poona, Pune, 1967

[3] Vasu, S.C., The Astadhyayi of Panini Vol I & II, Motilal Banarsidass, New Delhi, 1997

[4] Vasu, S.C., Siddhantakaumudi Vol I & II, Motilal Banarsidass, New Delhi, 1995

PHONOLOGY & SPEECH

Schwa Deletion and Syllable Economy in Indo Aryan Languages

Monojit Choudhury Anupam Basu Sudeshna Sarkar
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
PIN: 721302, INDIA
email: {monojit, anupam, Sudeshna}@cse.iitkgp.ernet.in

1. INTRODUCTION

Linguists propose new models for languages in order to explain language acquisition and processing by humans. Irregularities and exceptions to the theories are often explained by evidences from diachronic linguistics and other social and external phenomena. Absence of diachronic analysis in computational modelling of languages results in a large number of exceptions, which are commonly handled by *ad hoc* rules or exhaustive enumeration. These techniques lead to poor scalability and lack of graceful degradation of the systems along with increased complexity. Although complete modelling of the evolution of language systems is impossible due to the involvement of myriads of socio-political and cultural factors, it is definitely possible to model certain basic principles of *language change*.

In this paper we describe an algorithm for *schwa deletion* in Indo Aryan Languages (IAL) that is motivated by the diachronic evolution of the languages. The proposed computational framework models languages as a *constrained optimization system*, where a language evolves by optimizing the rate of communication, subjected to a set of constraints such as *ease of articulation* and *learning*, and *acoustic distinctiveness*. A *syllable minimization* based optimization function fitted to the aforementioned model has been used for solving the problem of schwa deletion with considerable success.

2. THE PROBLEM

Schwa is defined as the neutral middle vowel that occurs in unstressed syllables. The first vowel of the IAL alphabet {**a**} is the schwa. Normally, it is pronounced as /ə/ in Hindi and Sanskrit, and as /ɔ/ in Bengali. *Schwa deletion* is a phonological phenomenon where schwa is absent in the pronunciation of a particular word, although ideally it should have been pronounced.

Sanskrit and its derivatives, the modern IAL, are written from left to right using syllable-based scripts. All the vowels are explicitly represented using diacritical or non-diacritical marks around the consonant except for the schwa, which is the inherent vowel. Unlike Sanskrit, modern IAL like Hindi and Bengali allows deletion of schwa in certain contexts. **Table I** illustrates this phenomenon for the three languages. In order to determine the proper pronunciation of the words, it is necessary to predict which schwas are deleted and which are not. Thus, schwa deletion is an important issue for *grapheme-to-phoneme conversion* of IAL, which in turn is required for a good Text-to-Speech synthesizer [**1**].

Several theories have been proposed on the linguistic aspects of schwa deletion in Hindi [**2-3**] and its diachronic evolution [**4**]. Ohala [**2**] has summarized the rule for schwa deletion in Hindi as

$$\text{ə} \rightarrow \phi / \text{VC} \_\_ \text{CV}$$

Condition 1: There may be no morpheme boundary in the environment to the left.
Condition 2: The output of the rule should not violate the phonotactic constraints of Hindi
Convention: The rule applies from right to left

.	The Spelling	Pronunciation		
		Sanskrit	Hindi	Bengali
1	sāph**a**ly**a**	saphəlyə	sa.ɸəl.jə	ʃaɸol.l**o**
2	m**a**m**a**tā	məməta	məmta	məm**o**ta

Table I: Pronunciation of three different words in three different IAL. In Bengali {**a**} can also be pronounced as /o/ in certain contexts

The explanation of the rule was based on psycholinguistic evidences; diachronic facts were used only to explain the exceptions. Narsimhan *et al* [**5**] designed an algorithm for schwa deletion in Hindi based on this work. The reported accuracy of the algorithm is 89%. We do not know of any work on computational modelling of schwa deletion in Bengali.

3. SOUND CHANGE IN LANGUAGES

The fact that schwa deletion in IAL is a diachronic phenomenon has been substantiated in [**4**]. It can be inferred from the evidences cited in [**2**] that the motivation behind schwa deletion is *faster communication* through minimization of syllables. Some recent works on mathematical and simulation based modelling of language evolution [**6**] suggests that several features of languages emerge due to some basic cognitive and articulatory factors. According to them language can be modelled as a *multi-objective optimization system*, where the optimization criteria are

3a. **Minimization of effort** (in terms of energy and time spent while conveying a piece of information)

3b. **Minimization of learning time and effort**

3c. **Minimization of probability of misunderstanding** (in the sense of confusing one word with another)

These three criteria are mutually contradictory and therefore there exists no global optimum. Let us examine the phenomenon of schwa deletion under this multi-objective optimization model for language evolution. When a vowel is deleted from a word the number of syllables reduces by one and leads to faster communication. However, deletion of schwas in certain contexts might result in a consonant cluster which is very difficult to pronounce. This beats the very purpose of schwa deletion and therefore, is unacceptable.

There are contexts where deletion of schwa would not give rise to *inadmissible* consonant clusters. For example, in the Hindi/Bengali word **pari** (fairy, /pəri/ in Hindi), if the first schwa is deleted, the pronunciation would be /pri/, which does not violate the *phonotactic constraints*. The schwa, however, is not deleted, because /pəri/ and /pri/ are too distinct from each other to be interpreted as the same word. In this case, the deletion of schwa reduces the *acoustic distinctiveness* of the word from other words in the lexicon, which increases the probability of misunderstanding, and hence the schwa is not deleted in such a context.

4. COMPUTATIONAL FRAMEWORK

We propose the following diachronic explanation for schwa deletion in IAL.

> In Sanskrit none of the schwas are deleted. The modern IAL use the script and spelling conventions similar to Sanskrit. Due to a higher evolutionary pressure on the spoken form of the languages than on the written form, schwas are deleted in the pronunciation, but are still present in the graphemic forms. The deletion is a slow diachronic phenomenon, where in order to communicate faster, initially the speakers unknowingly deleted the schwas. Only those deletions were acceptable that did not lead to a syllable structure which was too difficult to pronounce, learn or understand. Gradually the number of people speaking the schwa-deleted form of the word outnumbered those who spoke the standard form and finally the former replaced the latter.

In this section, we describe a computational framework for modelling the aforementioned hypothesis based on the three optimization criteria stated in the last section. In the next section, we present an efficient algorithm for schwa deletion in IAL, which can be automatically constructed from this model, without the help of any other evidences.

4.1 Basic definitions

$\boldsymbol{\Sigma_g}$ ($\boldsymbol{\Sigma_p}$): A finite set of the graphemes (phonemes) in the language

$$\boldsymbol{\Sigma_g} = \boldsymbol{V_g} \cup \boldsymbol{C_g}, \qquad \boldsymbol{\Sigma_p} = \boldsymbol{V_p} \cup \boldsymbol{C_p}$$

Where

$\boldsymbol{V_g}$ ($\boldsymbol{V_p}$): Finite set of graphemes (phonemes), which are vowels

$\boldsymbol{C_g}$ ($\boldsymbol{C_p}$): Finite set of graphemes (phonemes), which are consonants. Semivowels are also considered as consonants.

$\boldsymbol{\alpha} \in \boldsymbol{V_g}$ is a special symbol, called schwa.

We define, $\boldsymbol{f_{g2p}}: \boldsymbol{\Sigma_g} \rightarrow \boldsymbol{\Sigma_p}$

$\boldsymbol{f_{g2p}}$ is the default mapping of the graphemes to the phonemes. This oversimplification is made here for two reasons. First, since IAL are mostly phonetic in nature, this in general is true and second, this assumption does not have any affect on the schwa deletion algorithm. A *word* $\boldsymbol{w}$ is defined as a 2-tuple $<\boldsymbol{w_g}, \boldsymbol{w_p}>$, where $\boldsymbol{w_g} \in \boldsymbol{\Sigma_g}^+$ and $\boldsymbol{w_p} \in \boldsymbol{\Sigma_p}^+$

A *grapheme-to-phoneme converter* is defined as a function $\boldsymbol{F_{g2p}}: \boldsymbol{\Sigma_g}^+ \rightarrow \boldsymbol{\Sigma_p}^+$, such that

$$\forall \boldsymbol{w} < \boldsymbol{w_g}, \boldsymbol{w_p} >, \boldsymbol{F_{g2p}}(\boldsymbol{w_g}) = \boldsymbol{w_p}$$

4.2 Phonotactic constraints

In order to model the *ease of articulation*, we start with the modelling of *phonotactic constraints*. A *consonant cluster* is a string of the form $\boldsymbol{C_p C_p}^+$. At the most generic level we can think of a *consonant cluster ranking* (***CCR***) function, (where $\mathbb{N}$ is the set of natural numbers) $\boldsymbol{CCR_p}: \boldsymbol{C_p}^+ \rightarrow \mathbb{N}$

The function CCR_p is independent of any language and every language has a threshold τ_{CCR}, such that a consonant cluster $x \in C_p^+$ is allowed in the language if and only if $CCR_p(x) \leq \tau_{CCR}$

Similarly, we can define two special variants of CCR_p – O_CCR_p and C_CCR_p, which ranks the *admissibility* of the consonant clusters at the *onset* and *coda* positions respectively with τ_{OCCR} and τ_{CCCR} as the corresponding threshold values.

The *sonority hierarchy* and *markedness conditions* point towards the existence of language independent ranking functions as hypothesized above. We define a Boolean function ADM that tell us about the admissibility of consonant clusters in a language.

$$ADM: C_p^+ \rightarrow \{0, 1\}, \text{ such that for } s \in C_p^*$$
$$(ADM(s) = 1) \Leftrightarrow (s \text{ is an admissible cluster})$$

In general, we can derive this function from CCR_p as

$$ADM(s) = \textbf{sign}(\tau_{CCR} - CCR_p(s))$$

However, we might have to forcefully convert some values to 0 due to accidental gaps.

4.3 Syllable and syllabification

We define a *syllable* σ_p as a regular expression, with the assumption that the *nucleus* contains a single vowel. Thus, $\sigma_p \in C_p^* V_p C_p^*$

The *syllabification* function SYL_p maps the phonetic representation w_p of a word w to a string of syllables $\sigma_{p1}\sigma_{p2}\ldots\sigma_{pm}$ such that the *efforts of articulation* and *learning* are minimum.

We model the *effort of articulation* using a *syllable ranking* function SR similar to CCR

$$SR_p : C_p^* V_p C_p^* \rightarrow \mathbb{N}$$

SR_p is mainly dependent on the *structure* of the syllable. We enumerate the first few terms of the function SR_p.

$SR_p(C_pV_p) = 1$, $SR_p(V_p) = 2$, $SR_p(C_pV_pC_p) = 3$, $SR_p(V_pC_p) = 4$, $SR_g(C_pC_pV_p) = 5$, $SR_g(C_pC_pV_pC_p) = 6$, $SR_g(C_pC_pC_pV_p) = 7$, $SR_g(C_pV_pC_pC_p) = 8$

Also, for any syllable σ_p,

$[O\_CCR_p(onset(\sigma_p)) > \tau_{OCCR}] \vee$
$[C\_CCR_p(coda(\sigma_p)) > \tau_{CCCR}] \Rightarrow (SR_g(\sigma_g) = \infty)$

We define a *syllabification* to be *valid* if all the syllables are *valid* (i.e. strings of the form $C_p^* V_p C_p^*$) and every symbol in the word is a part of one and only one *syllable*. We can define a partial ordering, $\leq_\sigma$, among the possible *valid syllabifications* of a given word based on SR_p such that the syllabification with smaller number of high ranked syllables is preferred to one that has more hard (high ranked) syllables. Now we define $SYL(w_p)$ as the set of all possible syllabifications $\sigma_1\sigma_2\ldots\sigma_m$ such that (i) $\sigma_1\sigma_2\ldots\sigma_m$ is a *valid syllabification* of w_p and (ii) there exist no other *valid syllabification* v of w_p such that $v \leq_\sigma \sigma_1\sigma_2\ldots\sigma_m$.

The definitions of syllable and syllabification are motivated by the *markedness conditions* and experimental results on child language acquisition [**7**], which show that some syllables and syllabifications are easier to learn and pronounce than others.

4.4 Acoustic distinctiveness constraints

Perceptual experiments show that speakers always articulate the onset of the syllables more clearly and correctly compared to the articulations of the vowel and the coda [**8**]. Therefore, it is likely that the hearer distinguish between syllables by paying more weight to the onset than to the coda. A continuous distance metric D_σ might be defined based on these experimental results, such that the *probability of confusion* (interpreting one syllable as another) between two syllables σ and σ' increases as the value of $D_\sigma(\sigma, \sigma')$ decreases. We can further define an *acoustic distance function* D_w using the function D_σ, which measures the probability of confusion between two arbitrary words in the phonetic domain.

In the case of schwa deletion, however, we want the *acoustic distance* between the ideal pronunciation and the normal pronunciation to be smaller, so that the word is not confused with other words in the lexicon. Formally, for the graphemic representation of a word $w_g = x_1x_2 \ldots x_n$,

$$D_w(f_{g2p}(x_1) f_{g2p}(x_2) \ldots f_{g2p}(x_n), F_{g2p}(w_g)) < \tau_{critical},$$

where $\tau_{critical}$ is the maximum allowable distance. Rather than modelling this as an optimization criterion, we reformulate this as a constraint. The simplification in this case serves our purpose.

We define, where $x \in C_p$

$$D_\sigma(x . f_{g2p}(\alpha), \phi) = 0 \quad \textbf{(4a)}$$
$$D_\sigma(\sigma_p, \sigma_p x) = 0 \quad \textbf{(4b)}$$

For all other cases D_σ is *infinity* (∞), unless the two syllables are identical. **(4c)**

(4a) allows the deletion of a schwa; **(4b)** allows the concatenation of a consonant at the coda position. **(4c)** restricts any change at the onset of a syllable or the vowels other than schwa.

On the basis of D_σ we can define $D_w(w_{g1}, w_{g2}) = \mathbf{0}$ if and only if there exists an alignment between the sequences $SYL(w_{g1})$ and $SYL(w_{g2})$, with possible *gaps* (ϕ or null syllables) such that for all the corresponding pairs of syllable taken from the two sequences, the *acoustic distinctiveness* (D_σ) between them is **0**. Thus, only operations allowed are deletion of a schwa and addition of a consonant at the coda position. Anything else is forbidden for the sake of *acoustic distinctiveness*.

4.5 The Algorithm

We want to define F_{g2p} for a language given ADM and D_w. F_{g2p} should be such that it enables faster communication by minimization of syllables by deletion of schwa. For this among all w_g' obtainable by deletion of some of the schwas from w_g, that respects both the ADM (*phonotactic*) and D_w (*acoustic distinctiveness*) constraints, the one with the minimum number of syllables is chosen as the output of F_{g2p}. It is easy to design an $\mathbf{O}(|w_g|)$ algorithm for this. (Please see [**1**] for details).

5. RESULTS & DISCUSSIONS

The algorithm was implemented for Bengali and Hindi and tested on a set of words. For Hindi the accuracy is 96% (without morphological analysis) and for Bengali it is 85% (with inflectional morphology information). For Hindi there was hardly any exception to the algorithm. For Bengali, the types of words that were incorrectly processed by the algorithm include a class of very frequently used, disyllabic modifier adjectives, certain suffixes, borrowed words from Sanskrit and compound words. In Bengali, the schwa which is retained (as opposed to the predictions by the algorithm) are pronounced as /o/ and not as / ɔ/. Since, /o/ is not a neutral vowel, deletion of /o/ is marked as compared to deletion of / ɔ/ which is unmarked. Transformation of schwa to some non-neutral vowel in Hindi is unknown and therefore, the algorithm works perfectly for Hindi.

Another interesting observation is that as we move towards south from northern India, we see that the tendency for schwa deletion reduces. Punjabi and Hindi have very high tendencies for schwa deletion. Bengali has a more restricted rules and schwa is deleted less frequently. In Oriya, schwas are rarely deleted. Similar patterns are observable in local dialects too.

6. CONCLUSION

The contribution of this paper is not just a better algorithm for schwa deletion, which is necessary for developing Text-to-speech synthesizers for IAL, but a new approach based on a *constrained optimization* framework, motivated by the diachronic evolution of languages. A closer look at the algorithm will reveal that it is not much different from the schwa deletion rule proposed in [**2**]. However, Ohala's rule was based on psycholinguistic and empirical observations, whereas we have derived the rule from a set of very basic assumptions (minimization of syllables and certain constraints). The algorithm itself can provide an explanation for the phenomenon.

Some of the questions that we would like to address in the future include modelling of optional schwa deletion in Bengali compound words, evolution of morpho-phonology for Bengali verb systems, and modelling of dialect diversity using diachronic clues. More realistic, yet manageable computational frameworks for holistic or detailed modelling of language evolution can also be an interesting area of future research.

REFERENCES

[1] Choudhury, M. & Basu, A. A Rule Based Algorithm for Schwa Deletion in Hindi. *Proc Int Conf Knowledge-Based Computer Systems*, Navi Mumbai, 2002 pp. 343 – 353

[2] Ohala, M. *Aspects of Hindi Phonology*, volume II. MLBD Series in Linguistics, Motilal Banarsidass, New Delhi. 1983.

[3] Kaira S. *Schwa-deletion in Hindi*. Language forum (back volumes), Bhari publications, 2 (1) 1976

[4] B. G. Misra 1967. *Historical Phonology of Standard Hindi: Proto Indo European to the present*. Cornell University Ph. D. dissertation

[5] Narasimhan, B., Sproat R. & Kiraz G.. Schwa-deletion in Hindi Text-to-Speech Synthesis. *Workshop on Computational Linguistics in South Asian Languages*, 21st SALA, Konstanz 2001

[6] Angelo Cangelosi and Domenico Parisi (Eds) 2002. *Simulating the Evolution of Language*. Springer-Verlag, London

[**7**] MacNeilage, P.F. and Davis B.L. On the Origin of Internal Structure of Word Forms. *Science*, 288:527-31, 2000

[8] Fosler-Lussier, E., Greenberg S. & Morgan N. Incorporating contextual phonetics into automatic speech recognition. *Proc. Int. Cong. Phon. Sci.*, San Francisco, pp. 611-614. 1999.

Bangla Pronunciation Rules and A Text-to-Speech System

Aniruddha Sen
School of Technology and Computer Science
Tata Institute of Fundamental Research
Homi Bhabha Road, Mumbai 400 005 INDIA
E-mail: asen@tifr.res.in

1. Introduction

Bangla is a major language not only in India: the world-over, it is fifth in the number of native speakers. In recent years, a range of Bangla language products were developed. The list includes word processors, spell checkers and also text-to-speech (TTS) systems.

As for Bangla TTS conversion, an area not adequately worked on is *text analysis* that generates pronunciation, given the input text and also (hopefully) determines the prosody nodes by language analysis. The pronunciation and the prosody information thus obtained may then be utilized to generate speech: by either *generative* or *concatenative* synthesis method.

Bangla, like most Indian languages, has *phonetic script*, in the sense that each elementary symbol represents one (or a combination of) phoneme(s). But there are many instances of: (a) deviations from the expected base pronunciation (usually in consonant clusters) and (b) ambiguities regarding the pronunciation of a phoneme (e.g. of a, e or s in different contexts). Some of these are systematic and may be captured by rules, but some others are irregular and are to be listed in exception dictionary.

In this paper, we deal with some key issues for Bangla TTS conversion. First we overview the scheme we employed (Sec. 2). Then we discuss formulation of pronunciation rules (Sec. 3) that is our present priority. We then discuss remaining tasks of text analysis (Sec. 4). For synthesis, we have adapted a formant synthesizer [1] for Bangla and this is discussed in Sec. 5. We conclude with the plans for near future, including an application that we are currently targeting.

To represent Bangla characters, we have used ITRANS convention of transliteration [2]. We represent the pronunciation too with the same convention, as there is an obvious symbol attached to most Bangla phonemes. An exception is the alternate pronunciation of 'e' (as in 'k*e*man' for Bangla or 'c*a*t' for English) that is represented by 'ae'. We represent a consonant or a vowel by C or V respectively.

2. Scheme to develop Bangla TTS system

The aim of our specific R & D was to develop a Bangla TTS system with application-level quality. We have decided to focus on key areas that will enable us to hook up a TTS system of reasonable quality within a moderate time frame and at the same time will enlighten us about the basic issues that will be instrumental for future refinements.

A very important issue for Bangla TTS system is formulating *pronunciation rules*. But, as we will show, *morphological analysis* is often needed to obtain correct pronunciation. Also, there are always enough *exceptions*, creating need for a *phonetic dictionary*. Making synthesis *natural-sounding* is a very important issue now. We must therefore formulate *prosody rules* too. And to detect stress nodes, a phrase/ clause level parser, working at least at semantic level, is helpful. All these are parts of *text analysis*. Out of these, we have singled out formulation of pronunciation rules as the key issue now. We have also taken up the task of building a large phonetic dictionary. With the help of these, and rudimentary morphological analysis system and prosody rules, we have hooked up the text analyzer that can be incrementally and modularly modified.

For *synthesis*, we are using a formant synthesizer for Indian speech sounds [1]. It is suitable for any Indian language in general, but has to be tuned to some specifics of the given language. We are adapting it to the specific phonetic and prosodic features of Bangla. We thus have a *complete system* to test the R & D results. This in turn speeds up the R & D process.

3. Formulation of pronunciation rules

As Bangla script is basically phonetic, each character usually has a default pronunciation. A major task in Bangla text analysis is to *modify default pronunciations* as per context by appropriate pronunciation (or *letter-to-sound*) rules. For implementation, we have classified these rules into consonant and vowel rules.

1. Systematic modifications of isolated Bangla consonants are: (a) Sanskrit/ Hindi semivowels *w* and *y* are converted to *b* and *j* respectively. In Bangla, y is represented by '*antashta-a*' and w, present only as allophones, is mostly represented as b even in computer. (b) Dental sibilant *s* is *often* and retroflexed sibilant *Sh* is *always* substituted by *palatal sh*. But in words derived directly from English, s is retained (e.g *s*Aikel, bA*s*).

2. Pronunciation can get modified in and around clusters of b, m, y, sh, Sh, s and h. The rules are often complex and depend on a number of contextual factors.

A notable change in Bangla is a cluster to geminate transformation (i.e. 'dwitwa'). It is observed in Cy, Cb, ksh, shm, sm and tm clusters, preceded by a vowel. For example: sa*ty*a (truth) → sha*tt*a, maha*tb*a (greatness) → maha*tt*a, a*kSh*ata (unharmed) → a*kkh*ata, aka*sm*At (suddenly) → aka*shsh*At, grI*Shm*a (summer) → grI*shsh*a, A*tm*A (soul) → A*tt*A. However, word-initially or after a consonant, the second consonant is just dropped. For example, *dy*uti (glow) → *d*uti, aga*sty*a (a famous sage) → aga*st*a, *db*a*ndb*a (conflict) → *dand*a, *kSh*ati (loss) → *kh*ati, *sm*aran (remembrance) → *sh*aran.

For CyA, *A* is often modified to *ae*, e.g. bikh*yA*ta (famous) → bikh*khae*to. It also may happen in bya, e.g. ab*ya*basthA (disarray) → ab*bae*basthA.

Overall, whether C_1C_2 → C_1C_1 or A → ae changes take place depends on whether the cluster is : (a) word-initial (b) after a C or V (c) after a prefix (d) after a root word (e) after H (visarga) or (f) after an 'a' that will be deleted. We have noted alternate possibilities for each cluster, formed rules in each of the above mentioned positions and have also recorded the exceptions.

As for h-clusters, hm and hn changes to mh and nh in pure pronunciation and to mm and nn colloquially (e.g. bra*hm*A (a God) → bra*mh*A or bra*mm*A; aparA*hn*a (afternoon) → aparA*nh*a or aparA*nn*a). In hb cluster, b is changed to labial fricative *v* (as in 'van') and the h is modified to *u* or *o*, depending on the height of the preceding vowel (e.g. j*ihb*a (tongue) → j*iuv*a, but *Ahb*an (a call) → *aov*An). An hy cluster is substituted by jjh geminate (e.g. sa*hy*a (toleration) → sa*jjh*a).

Another consonant rule is regarding a sibilant-consonant cluster. If the consonant is r, l or n, s does not undergo the usual s → sh change, but sh and Sh are often changed to s. There are regional and person-to-person variations in this aspect.

3. *Visarga* (H), classified as a separate group in Sanskrit/ Bangla grammar, has been clubbed with consonants for convenience. Word-finally, H → h. Within a word, H can either cause gemination (e.g. mana*HkSh*unna (disappointed) → mana*kkh*unna), or may just add an s (e.g. mana*Hk*AmanA (desire) → mana*shk*amanA). It doesn't occur word-initially.

4. Pronunciation of neutral vowel ('a') in Bangla is ambiguous. 'a' may either be silent (schwa deletion) or may be pronounced either as usual 'a' (as in 'b*a*ll') or as 'o' (as in 'g*oa*l'), e.g., 'samara' (war) → shamor. Here, the 1st 'a' is pronounced as 'a', the 2nd one as 'o' and the 3rd one is silent. To find rules to cover all occurrences is not feasible now and unlikely in future. But some ground can be covered by a few simple criteria such as: (a) a → o, if next vowel is i, I, u or U, else a → a (e.g. g*a*ru (cow) → g*o*ru, but t*a*ral (liquid) → t*a*raol). (b) In a mono-syllabic C_1a C_2 word, a → o, if C_2 is n or N and a → a otherwise, e.g. m*a*n (mind) → mon, but phal (fruit) → ph*a*l. (c) In Bangla, an 'a' after a consonant cluster is retained, e.g. in suka*nta*. An end-of-word 'a', if retained, is pronounced as 'oh'. (d) After a Cr cluster, *a* is usually changed to *o*, e.g. *kra*mAgata (continuously) → *kro*mA gata. However, if a *y* follws, a → a after kr and tr, but a → o after pr (e.g. pr*a*yog (application) → pr*o*yog, but kr*a*y (purchase) → kr*a*y).

In Bangla, 'e' (as in 'gr*ey*') is sometimes pronounced as 'ae'. Again, rules are inadequate to cover all instances, but a few may be cited: (a) e → ae, if the next vowel is a or A. Else, it remains the same. For example, k*e*man (how) → k*ae*man, but b*e*ti (daughter) → b*e*ti. (b) However, in words derived from Sanskrit ('tatsama'), 'ey' remains unchanged, e.g. in b*ey*danA (pain).

Among other vowels, e/E and u/U are yet maintained in Bangla spelling, but are not differentiated in pronunciation. An elongation of e or u indicates accent and not phonation in Bangla [3]. It may also be noted that ai and au, pronounced as *vowels* 'ae' and 'aw' in Hindi, are realized as *dipthongs* in Bangla.

The total rules and sub rules are many more. An excellent treatise is available in [4] that we have extensively used as the base of our rule formulation research. One may also refer to [3] and [5]. But we don't expect these to cover the whole space, at least now. Even consonant rules may remain inadequate, as often the word-initial rules get applied *selectively*

across morph boundaries within words. For example, *sb*arga (heaven) → *sh*arga and *sb*Ami (owner) → *sh*Ami. But whereas bhu(-)sbAmi (landowner) → bhu*shsh*Ami, bhu(-)sbarga (heaven on earth) → bhu*sh*arga (no germination). Thus, for both vowel and consonant pronunciation, rules must be supplemented by morphological analysis and exception dictionaries. Table 1 shows an example of general rules for Cy.

Context	Cy-Rule	If A follows
Word initial Cy	Cy → C	A → ae
Cy after a prefix	Cy → CC	A → ae
Cy after a root word	Cy → C	A → ae
Word-mid VCy	Cy → CC	A → A
Word-mid CCy	Cy → C	A →A
Cy after 'visarga' (H)	Cy → CC	A → ae
ry	Cy → Cy	A → A

Table 1 Gemination and Vowel Change Rules for Cy

4. Text analysis: the remaining tasks

(1) The next prioritized task was creating *phonetic dictionary*. We started with an electronically available list of about 55,000 Bangla words in ISCII (courtesy: Webel Mediatronics Ltd.). Default pronunciations were first generated by existing letter-to-sound rules and are then being gradually modified manually. An existing default helps the process, as only the known areas of ambiguities (e.g. schwa, germination in a cluster) are to be changed. The exception words are entered first. The dictionary has a 'word type' field that represents a parts-of-speech type tag and is filled manually along with the pronunciation. There is also a field to indicate that this word has been taken care of and should not be disturbed when the default pronunciations are changed next by rule program. An (assumed) example of an entry is:

bikShata bikkhaeto 3 y.

Here, 1st field is the spelled word in ITRANS, the 2nd is the pronunciation (with convention used in this paper), the 3rd field indicates a type 3 (presumably, a noun) and 'y' in the 4th is a flag to the rule program to leave it alone. This is from the ITRANS-version of the dictionary. We have made an ISCII version too.

We have also developed an intelligent search engine to list occurrences of phoneme clusters in *various contexts* (i.e. in word-initial/ mid/ final position or before/ after vowels/ consonants) in a file. Manual analyses on the lists are done to decide rules and exceptions. The search-engine can create sub-lists, too.

As for other modules, we are now mostly on research and design stage. In morphological analysis, we implemented some minimum needful that influences 'dwitwa', as discussed before. We have put a type tag to the standard prefixes ('upasarga's, e.g. pra, para, apa, sam, aba) in the dictionary. We can also identify a 'root' before a possible point of germination, if that is present in the dictionary. Thus, we are utilizing the large word-list, without relying on the dictionary pronunciation. But for schwa deletion and decision of a/o, e/ae, full-fledged morphological analysis is needed. The issues of removing suffix (that modifies root) and handling ambiguities are being currently worked on.

Another vital issue is prosody. We believe that this will remain an open problem for years to come, whereas the other issues may settle soon. We have implemented an elementary system that creates 'markers' to alter pitch, intensity and duration at some qualitative levels. The decisions are based on crude analysis of the type of sentence (assertive, interrogative or exclamatory) and on the elementary knowledge that in Bangla, stress is decided more by *position* than by *meaning* [3].

5. Phoneme-to-speech synthesizer

The phoneme-to-speech module of the TTS accepts the phoneme string and stress markers from the front-end text analyzer and converts them to digitized speech. It is the basic TIFR formant synthesizer [1] adapted for Bangla pronunciation. The synthesizer uses a set of rules, specific to the Indian phoneme set, to generate acoustic-phonetic parameters (including formant frequencies). A Klatt-type source-filter model is used to generate speech, corresponding to the parameters. The synthesizer covers all typical Indian phonemes, including retroflexed, dental and aspirated consonants and nasalized vowels. It is already used to generate Hindi, Indian English and Marathi speech.

Some changes done for adapting the synthesizer to Bangla are: (a) Formant frequencies of the neutral vowel are different in Bangla (in comparison to the corresponding values for Hindi and Marathi). It is realized as either a short back vowel (F1 = 600, F2 = 1000) or a short rounded vowel (F1=450, F2 = 900). In Bangla, i/I, u/U are not differentiated. We, therefore, used only one (near-short) form. (b) Palatal (talavya) 'sh' in Bangla is different from Hindi 'sh' in spectral energy distribution as well as in manner of articulation. This is being researched on and the results applied. (c) Retroflexed 'n' (N) is not used in Bangla in general. It is only realized as an 'allophonic' variation of 'n' in retroflexed stop-nasal cluster, e.g. in 'ANDA' (egg). Such clusters are realized as a special phoneme type in the synthesizers (like a separate phoneme-type). So, 'N' may be removed from its

repertoire. (d) Bangla, has a flapped 'r' (daye-shunno ra) that is realized almost identically in Hindi and Bangla. But Bangla also has a 'dhaye-shunno ra', an *aspirated* flapped ‘r’. It is realized as flapped d plus h now. For better synthesis, it must be treated as a separate, single phoneme.

The synthesizer can accept stress markers (for pitch, duration, intensity) generated by the front-end text processor and change the parameter tracks accordingly.

Table 2 lists formant frequencies used for vowels.

Vowel	F1 (Hz)	F2 (Hz)	F3 (Hz)	Duration (ms)
a	600	1000	2500	65
short o	450	900	2400	65
A	700	1200	2600	110
I	350	2200	2800	90
U	350	850	2300	90
e	450	2000	2700	110
o	450	900	2400	110
ae	650	1800	2700	110

Table2 Formant Frequencies and Durations of Vowels.

6. Conclusions

We have presented an R & D effort, aimed at building a quality Bangla TTS system in limited time. Initially, we have singled out areas such as formalizing quality pronunciation rules, forming a large phonetic dictionary and adapting an Indian language phoneme-to-speech synthesizer to generate quality Bangla speech. We have built a system that can put to use the results of the R & D at any instance. The system, dictionary and other language tools developed help to speed up the R & D work.

Morphological analysis and prosody are the vital areas to be stressed next. We are designing the morphological analyzer and making schemes that suits suffix separation and ambiguity handling best. For prosody, we have planned to record and analyze a bulk of Bangla sentences (preferably from TV channels) to find the basic patterns and then formalize the modulations by rules. We intend to capture gross prosody patterns first and then refine them gradually.

Another area is the routine handling of special types of inputs such as numerals, abbreviations, acronyms, date, time, currency etc. Identification and processing of such inputs are not difficult in Bangla [5] and will be completed in a routine manner.

The system being developed is general-purpose and may be put to any relevant use. But a specific application currently in mind is to augment the capabilities of an existing computer system, developed by Webel Mediatronics Ltd., to train the blind. Presently, there is provision to verify text typed (in Braille) character-by-character. Synthesis at sentence level will enable the users to read stored files with ease.

Indian language dictionaries don’t indicate pronunciation, presumably with the belief that it is obvious! As we have shown, that is not true at least for Bangla (and even not true for Hindi, Marathi and so on). Standard phonetic dictionaries, in both printed and electronic forms, are the needs of Indian languages and the usefulness will go beyond the needs of Text-to-speech or speech-to-text systems.

7. Acknowledgement

I am grateful to Prof. G. Sengupta, Hyderabad University for the rule base and Mr. S. Goswami, Webel Mediatronics Ltd. for the word list.

References

[1] X.A. Furtado and A. Sen, ‘Synthesis of unlimited speech in Indian languages using formant-based rules’, *Sadhana*, 21(3), pp. 345-362, 1996.

[2] ITRANS web site: http://www.aczone.com/itrans/

[3] S. Sen and S. Sen, *Vyakaran Praveshika*, A.K. Sarkar Publ. Pvt. Ltd., Kolkata, 1989.

[4] P. Dasgupta and G. Sengupta, ‘The Bangla-Asamiya script and its representation in Unicode’, ‘Working papers of intl. symposium on Indic scripts: past and future’ (ed.: P. Bhaskararao), ICLAA, Tokyo, Dec. 2003.

[5] A. Sen, ‘A text analyzer for Bangla text-to-speech synthesis’, Proc., CODEC-2004, Kolkata, Jan 2004, paper no. adc_0404_CO.

A NOVEL APPROACH FOR DESIGNING A SPEECH SYNTHESIS SYSTEM FOR INDIAN LANGUAGES

Sanghamitra Mohanty *sangham1@rediffmail.com*

Suman Bhattacharya *suman1_bh@yahoo.com*

Sumit Bose *mail_sumit_bose@rediffmail.com*

RC-ILTS-ORIYA
Post Graduate Dept. of Comp. Sc.&Appl^n
Utkal University, Bhubaneswar
Orissa-751004, India

Abstract

Designing a speech synthesizer for Indian languages do not have a long history, rather it started a decade back. Different theoretical approaches have been put forward in this regard. We are designing the Text-To-Speech system for Oriya, Hindi and Bangla. The present study made by us is based upon the rules of ancient Indian philology derived from Paniniyashikshya and Pratishakshya. While designing, prosody plays a major role and unless the speech signal is properly parameterised the proper naturalness cannot be realised. Hence we have obtained eighteen odd different parameters of the signal for designing a robust TTS system. Some of the example of the parameter extraction is explained in this paper.

As we know the Indian languages have Sanskrit origin, we have already given an approach to design a robust speech synthesis system in our earlier papers. We have already applied these techniques on Oriya language [1][4]. In this paper we focus on how to obtain some parameters based on Paninian philology, which will help us to design a robust synthesis system. The signal parameter estimation is really a daunting task for any speech researcher. In our case we have tested for Oriya language after meticulous study on the rules of Panini. The exquisiteness of our language is that each of the characters is uttered independently. The method we have adopted for Oriya language is character-based concatenation. For other Indian languages such as Hindi and Bangla, character-based approach is not viable and a level of naturalness cannot be consummated. Here syllable-based approach is efficacious. We have recorded the words from naturally speaking persons for different languages. The words are split into characters and syllables as per the requirement for a particular approach. Combination of characters sometimes makes conjuncts where the duration is an important factor. For Oriya language the characters are segregated into pure consonants and vowels. As we know the vowels are the most interesting class of sound in any language and they play a major role in the process of utterance of a particular character. The duration of vowels are properly studied for their location in a word as per the requirements. We maintain the acoustic value of the wave file for the phone and diaphone. These are merged as per prerequisite to generate the modified consonants influenced by *mAtras*, *phalAs*, *yuktAs* etc. The data we obtain from the wave file is now processed for parameter estimation. The microphone used in the process of recording normally introduces undesired side effects such as hum, loss of low and high frequency information and non-linear distortion etc. The sharp attenuation of low and high frequencies often causes problems for the subsequent parametric spectral analysis algorithm. Normally we use the sampling frequency of 12 kHz, 16 kHz or 22 kHz for the processing purpose for better time and frequency resolution. In this stage the signal with high SNR is taken into consideration. We use *Kalaman* filter for this purpose. As the voiced section of the speech signal naturally have a negative spectral slope of approximately 20dB per decade due to psychological characteristic of the speech production system, this filter serves to offset this natural slope before spectral analysis and hence it improves the efficiency of the spectral analysis for parameter estimation. It has been observed that the pre-emphasis filter raises the frequency above 5kHz, where the auditory system becomes increasingly less sensitive. The frequencies above 5kHz are naturally attenuated by the speech production system and a smaller weight being assigned to the frequency above it.

Out of the parameters at the outset we bring our all endeavor on the most important one and that is the fundamental frequency (f_0). This is required to determine the pitch for application to prosody. The algorithm that we have considered operates on the cepstrum of the speech signal. F_0 is often processed in a logarithmic scale rather than a linear scale to match the resolution of the human auditory system. We can define f_0 as:

$$f(n) = \log (f_0 (n)) \quad (1)$$

here *n* represents the discrete time.

It has been observed that for voiced speech the value of f_0 lies in the range from 80Hz to 500Hz. For unvoiced region f_0 is undefined.

The pitch pattern or fundamental frequency over a sentence in natural speech is a combination of many factors. The pitch contour depends on the meaning of the sentence. If we look at a normal speech, the pitch slightly dwindles towards the end of the sentences. When the sentence is in question form, the pitch pattern will raise to the end of the sentence [Figure 1(a), 2(a), 3(a)] [6]. If in the end of the sentence there may also be a continuous rise, which indicates there is more speech to come [Figure 1(b), 2(b), 3(b)] [6]. A raise or fall in fundamental frequency can also indicate a stressed syllable. Finally the pitch contour is also exaggerated by gender, physical and emotional state and attitude of the speaker.

By using the cited Paninian rule [1][4] we can investigate the duration or time characteristics at several levels from phoneme (segmental) durations to sentence level timing, speaking rate and rhythm. Usually some inherent duration for character is modified by rules between maximum and minimum durations. To cite some examples the consonants in non-word initial positions are shortened, emphasized words are significantly lengthened, or stressed vowel or sonorant preceded by a voiceless plosive is lengthened. In general the character duration differs due to neighbouring characters. At sentence level, the speech rate, rhythm and the correct placing of pauses for correct phrase boundaries are important. With some method to control duration or fundamental frequency, such as the PSOLA,ENSOLA etc, the manipulation of one-feature affects to another is determined [2].

The intensity pattern is perceived as the loudness of speech over the time. At syllable level the vowels are usually more intense than consonants and at phrase level the syllable at the end of an utterance can become weaker in intensity. The power contained in the signal at a particular instant is given by

$$P(n) = 1/Ns * \sum_{m=0}^{Ns-1} (w(m)*s(n-Ns/2+m))^2 \qquad (2)$$

Here *Ns* determines the number of samples used to compute the power and *w(m)* denotes the weighing factor. Also it is seen that the intensity of the speech signal is dependent on the fundamental frequency and the relationship is that the intensity of voiced sound goes up in proportion to the fundamental frequency (Klatt 1987). Power is normally computed on a frame basis. The frame duration we represent here is T_f and it is defined as the length of the time over which a set of parameters is valid. In practical system the frame duration is 10 ms to 20 ms. The frame duration and the window duration together control the rate at which the power values track the dynamics of the signal. Since a shorter frame duration is used to capture rapid dynamics of the spectrum, the window duration should also be correspondingly shorter so that the detail in the spectrum is not excessively smoothed. The amount of overlap to some extent controls how quickly parameters can change from one frame to other. The percentage of overlap is given by:

$$\% \ Overlap=(T_w - T_f) / T_w * 100\% \qquad (3)$$

T_w and T_f are respectively the window duration and frame duration.

Out of the several spectral analysis transform technique we have taken the eigen-vector method and calculated the main lobe of frequencies as well as the critical bandwidth of the speech signal for different formants using the formula:

$$BW_{critical} = 25 + 75* [1+1.4(f/1000)^2]^{0.69} \qquad (4)$$

The bandwidth is measured in *Mel* scale [5].
Computation of the parameters is the major task using the several popular algorithms available.

Incorporating prosody is an important task in any speech synthesis system. The prosody or suprasegmental features consists of pitch, duration and stress over the time. Prosodic features can be divided into several levels such as character, word or phrase level. At word level vowels are more intense than consonants[3]. The first and foremost thing in our approach is to record the voice of a person who has good command over the utterance. The voice categories affected by the emotions are usually categorized and the parameters are stored for the processing phase. The voice quality contains largely constant voice categories such as loudness and breathiness. The breathiness is normally marked in angry voice. The pitch contour and its dynamic changes carry important emotional information both in the general form for the whole sentence and in small fluctuations at word and phonemic level. The time characteristics contain the general rhythm, speech rate, the lengthening and shortening of the stressed syllables, the length of the content words and the duration and placing of pauses. The anger in speech causes intensity with dynamic changes. The voice is very breathy and has tense articulation for content words [Figure 1(c), 2(c), 3(c)]. In happiness condition though the intensity is of increasing trend, the voice is light and breathy [Figure 1(d), 2(d), 3(d)] [6]. In fear or anxiety the intensity of speech is lower with no dynamic changes. The energy at lower frequency is also reduced. The average pitch and pitch changes are slightly larger than the neutral speech. When we talk about whispering, it is produced by speaking with high breathiness without

fundamental frequency. The tiredness causes a loss of elasticity of articulatory muscles leading to a lower voice and narrow pitch range.

We have stressed on the parameters which aims to prosodic analysis of speech signal and hence can be incorporated into the concatenation approach of speech synthesis to bring out naturalness.

References

[1] S.Mohanty, S.Bhattacharya & A.K.Senapati,*"Classification of vowel on the basis of utterance for phoneme recognition*

[2] *in Indian languages in the context of Sanskrit."*, Frontiers of Research on Speech and Music, FRSM-2004.

[3] Picone Joseph, "*Signal Modeling Techniques In Speech Recognition*", Proceedings of IEEE, 3rd June'1993.

[4] S.Mohanty, S.Bhattacharya & A.K.Senapati,*"An approach to phoneme recognition in the process of speech recognition of Indian languages Oriya: A case study."* Indian Science Congress Association' 2004.

[5] Whitney W.D,"*Sanskrit Grammar*" Motilal Banarasisdass, Delhi, 2002

[6] Rabiner L., Juang B.H., *"Fundamentals of Speech Recognition"*, Prentice Hall, New Jersey'1993.

[7] Snack Tool Kit- ver. 5

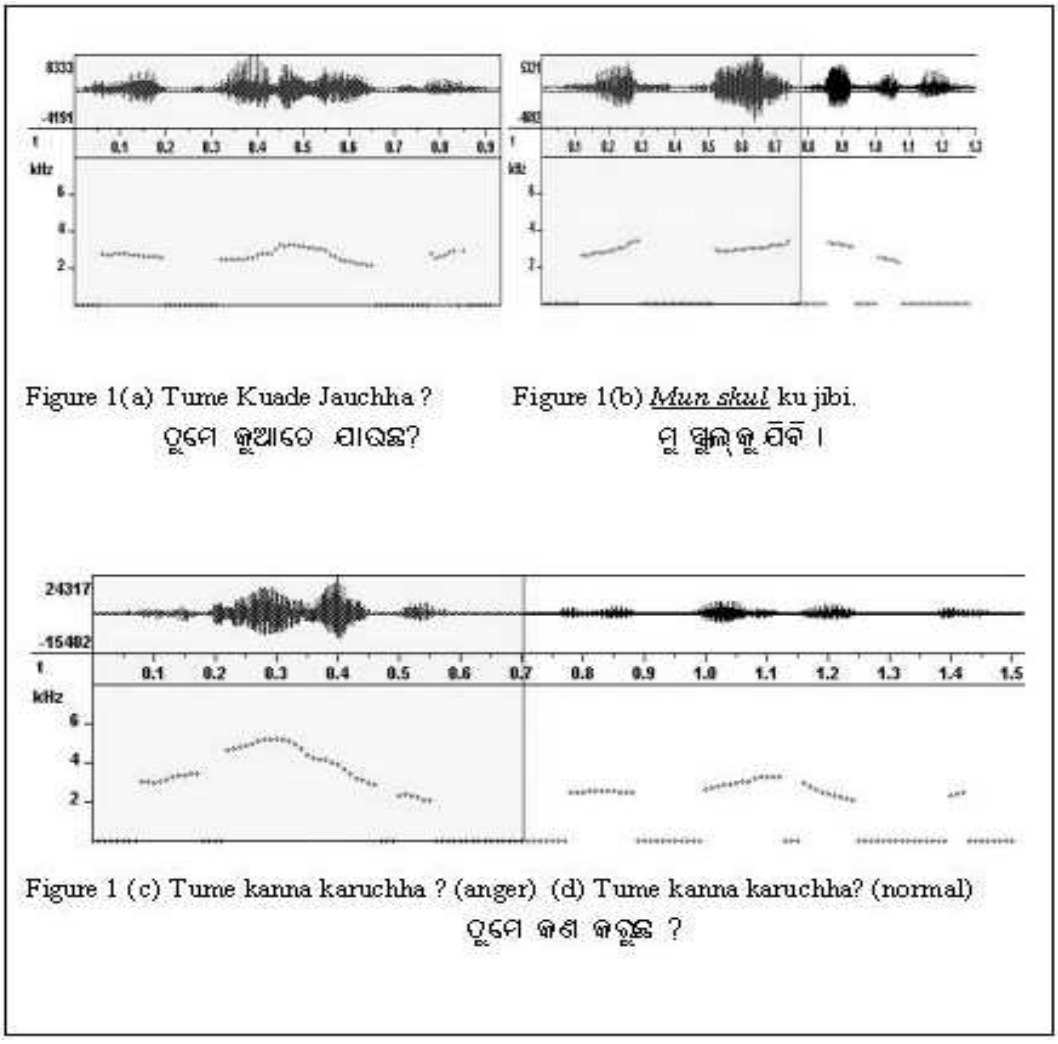


Figure 1(a) Tume Kuade Jauchha ?
ତୁମେ କୁଆଡେ ଯାଉଛ?

Figure 1(b) *Mun skul* ku jibi.
ମୁ ସ୍କୁଲ୍ କୁ ଯିବି ।

Figure 1 (c) Tume kanna karuchha ? (anger) (d) Tume kanna karuchha? (normal)
ତୁମେ କଣ କରୁଛ ?

Different Pitch variation For Oriya

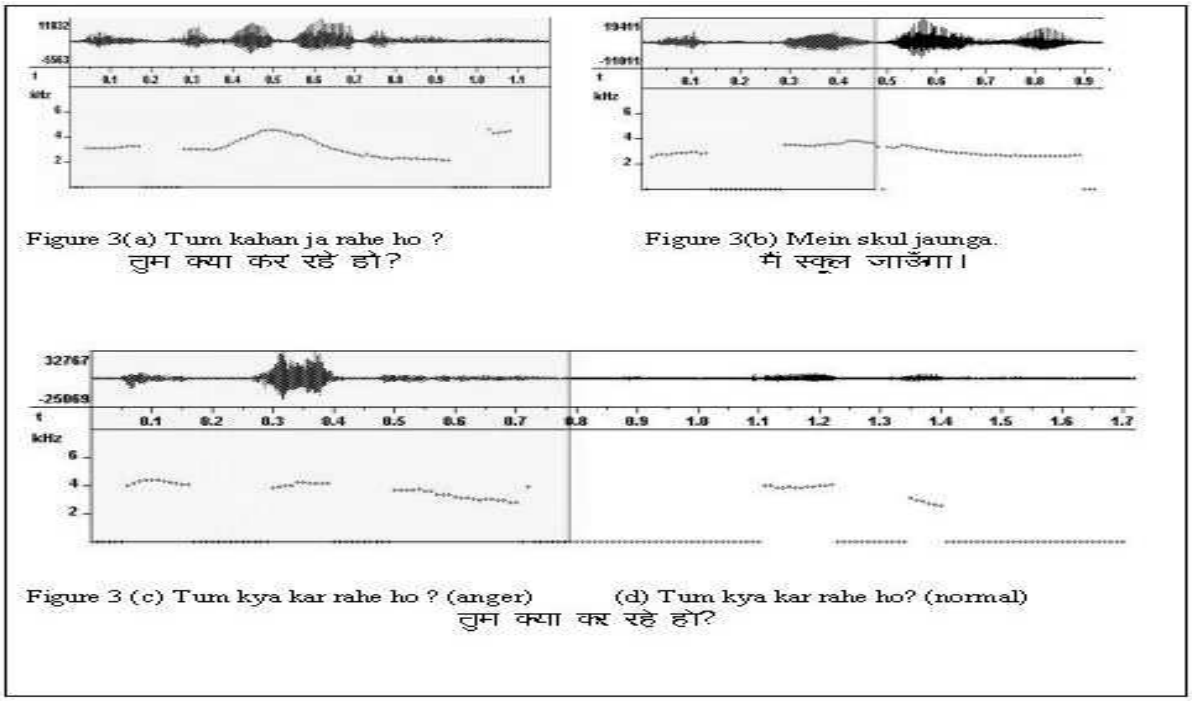


Figure 3(a) Tum kahan ja rahe ho ?
तुम क्या कर रहे हो?

Figure 3(b) Mein skul jaunga.
मैं स्कूल जाऊँगा।

Figure 3 (c) Tum kya kar rahe ho ? (anger) (d) Tum kya kar rahe ho? (normal)
तुम क्या कर रहे हो?

Different Pitch variation For Hindi

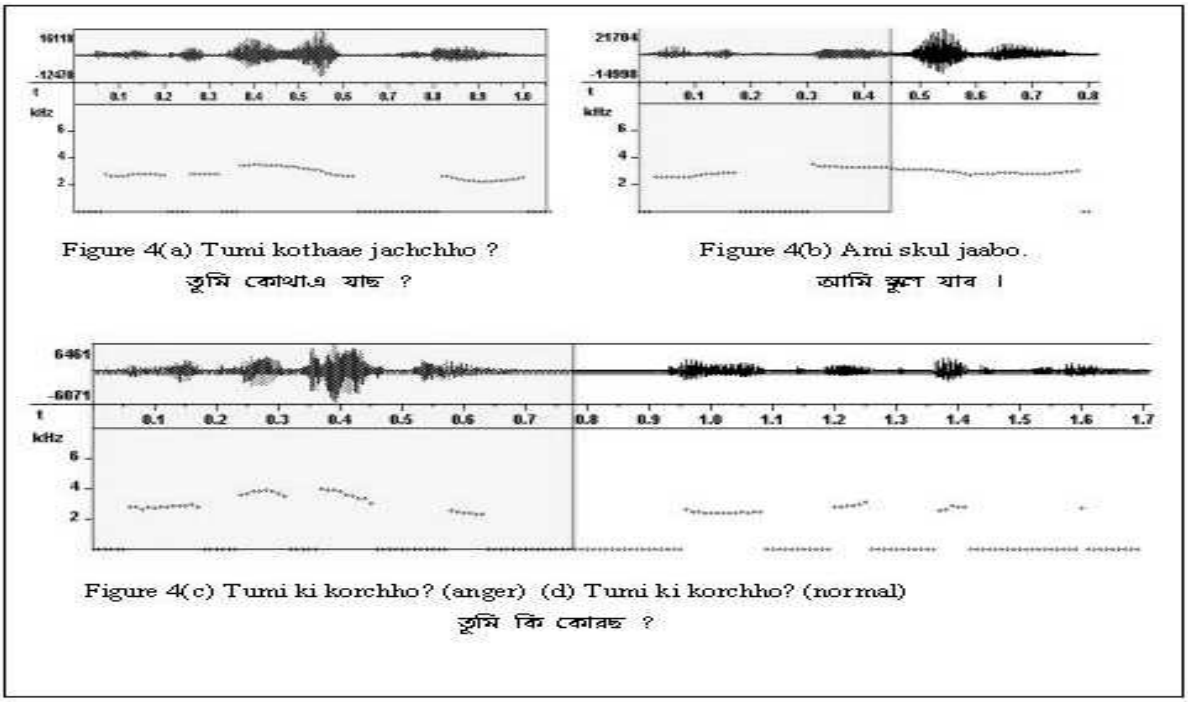


Figure 4(a) Tumi kothaae jachchho ?
তুমি কোথাএ যাচ্ছ ?

Figure 4(b) Ami skul jaabo.
আমি স্কুল যাব ।

Figure 4(c) Tumi ki korchho? (anger) (d) Tumi ki korchho? (normal)
তুমি কি করছ ?

Different Pitch variation For Bangla

Acknowledgement

We acknowledge the Ministry of Communication and Information Technology, Govt. of India for their financial support in persuasion of this work.

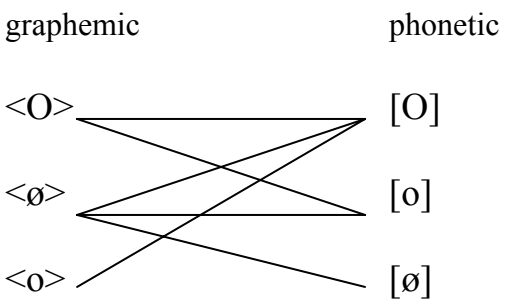
The Story of <O>: A Phonetic Dilemma

Mina Dan
Dept. of Linguistics
University of Calcutta

Unlike the other symbols <O>, the first letter of the Bangla alphabet, has a unique story – it is one of the vowel symbols as well as the inherent vowel with each consonant symbol unless otherwise discarded. Thus it has two graphemic representations, viz. the grapheme <O> and zero or <ø>.

At the phonetic level it has three values, viz. [O], [o] and [ø], among which [o] is also the phonetic value of another vowel symbol <o>. In other words, the phonetic [o] is represented by two graphemes.

The correspondence between the two levels, viz. graphemic and phonetic , could be shown as follows:



Thus the two graphemic representations, three phonetic values and sharing of the same phonetic value with another grapheme – all these facts together throw great challenge to the fields of phonology, language teaching, and NLP, and in NLP especially to the task of phoneme-to-grapheme / grapheme-to-phoneme conversion of the letter <O>.

In this paper I shall restrict myself within the domain of grapheme-to-phoneme conversion of the letter <O>. The cases that come under it are listed below.

The grapheme <O> and its variation <ø>, henceforth <O/ø>, occur word initially as well as in the initial syllable. Both the positions I call initial position. Initially <O/ø> has two phonetic readings, viz. [O] and [o], e.g.

1. <O/ø>=[O]
   Onek ‘many’ Oto ‘that much’
   kOmol ‘lotus’
   SOrol ‘simple’ pOTol ‘a vegetable’
2. <O/ø>=[o]
   notun ‘new’ oti ‘very’
   dolil ‘deed’
   orjun ‘a name’ moru ‘desert’
   prothom ‘1st’

Word finally only the variation <ø> occurs, which has [o] and [ø] phonetic values, e.g.

3. <ø>=[o]
   onno ‘other’ gOlpo ‘story’
   Soto ‘hundred’
   asto ‘whole’ gOrto ‘hole’
   nOSTo ‘spoilt’
4. <ø>=[ø]
   jOl ‘water’ akaS ‘sky’
   nOrom ‘soft’
   deS ‘country’ bagan ‘garden’
   gobhir ‘deep’

Word medially <O/ø> has all the three phonetic readings, e.g.

5. <ø>=[O]
   bigOto ‘previous’ ujjOl ‘bright’
   urbOr ‘fertile’
   akOrno ‘till ear’ OSobOrno ‘inter caste’
   OnobOroto ‘continuously’
   biSOY ‘matter’ OnorgOl ‘non-stop’
   OnolOS ‘not lazy’
6. <ø>=[o]
   OloS ‘lazy’ alocona ‘discussion’
   gOrom ‘hot’
   OboSOr ‘leisure’ Obodomito ‘suppressed’
   bhiSon ‘extremely’

7. <ø>=[ø]

Ekmatro ‘sole’ hatkORa ‘ koltOla ‘bathroom’

In monosyllabic words word final <ø> is always [O], but the penultimate <ø> shows two readings, viz. [O] and [o], e.g.

8. <ø>=[O] finally

thO ‘astonished’ SO ‘hundred’

9. <ø>=[O]

pOth ‘way’ mOt ‘opinion’ rON ‘colour’

jOr ‘fever’ SOr ‘cream of milk’ kOl ‘tap’

10. <ø>=[o]

mon ‘mind’ bon ‘forest’

dhon ‘property’ jom ‘god of death’

Various phonological cross currents result in different phonetic readings of <O/ø>. These cross currents are the actual key factors responsible for the ‘correct’ phonetic readings of <O/ø> in various environments. For example, a set of phonotactic rules and a set of phonological rules work hand in hand to determine the correct phonetic reading of <O/ø>. A few such rules are as follows:

11. Rule of word boundary

This rule says that the <O/ø> of the syllable immediately after a word boundary is [O], e.g.

#Onek# #a#kOrno#

#SOrol# #pOTol# etc.

12. Rule of assimilation

The <O/ø> followed by an <i/i:> or <u/u:> in the next syllable is [o], e.g.

oti, orjun, dolil, notun, moru, etc.

13. Rule of dissimilation

The <O/ø> preceded by an <i/i:> or <u/u:> in the preceding syllable is [O], e.g.

ujjOl biSOY urbOr etc.

14. Rule of stress shift

If the output of the rule of word boundary has two consecutive <ø>s in 2nd and 3rd syllables then they will be phonetically [o] and [O] respectively, e.g.

OnobOroto, OnolOS, OSobOrno etc.

15. Word final rules

i. Word finally the <ø> is [o] if immediately preceded by <CC>, i.e. a conjunct character, e.g.

gOlpo, nOSTo, gOrto etc.

ii. Generally the final <ø> immediately preceded by <t> is [o], e.g.

bigOto, jibito ‘alive’, Ononto ‘endless’ etc.

iii. Otherwise the word final <ø> is [ø], e.g.

akaS deS bagan gobhir etc.

16. Character specific rules

The <ø> immediately preceded by <pr> is [o], e.g. prothom prodhan ‘chief’ proSno ‘question’ etc.

17. Rules of monosyllables

i. Word final <ø> is [O], e.g.

thO SO etc.

ii. Non-final <ø> is [O] if followed by final non-nasal, e.g.

jOl pOth mOt

jOr kOl etc.

iii. Non-final <ø> is generally [o] if followed by a final nasal, e.g.

mon bon jom dhon etc.

These key factors, therefore, should be efficiently accommodated in a computational package of NLP.

The two basic aspects of this task are precisely WHERE and HOW to accommodate these key factors.

This paper, from a linguistic point of view, on the basis of empirical data, analyses the phonological factors, identifies the phonological processes, formulates the rules and specifies the order thereof. Moreover, depending on the nature of the change, character of the environment and type of the lexical items it suggests that these key factors may be accommodated in a scattered manner in different modules of the NLP system rather than in a single one and all these modules jointly will take care of the grapheme-to-phoneme conversion task.

The question HOW, i.e. how to encode these linguistic information and assumptions in terms of a meta language, belongs to the interdisciplinary plane between linguistics and computer science. In this respect the present paper provides an open area and looks forward for a hand from the other end.

PARTS OF SPEECH & SEMANTIC ROLE TAGGING

Semantic Role Tagging Using FrameNet

Ankit Anand, Gaurav Pandey, Amitabha Mukerjee
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur, India
{ankanand,gpandey,amit}@cse.iitk.ac.in
Achla Raina
Department of Humanities and Social Sciences,
Indian Institute of Technology, Kanpur, India
achla@iitk.ac.in

I. Introduction

Semantic roles [1] are defined as the underlying relationship that a constituent has with the main verb in a clause. For example, in the sentence

$$[_{e_1} John\ saw\ [_{e_2} Bill\ hitting\ Mary\ with\ a\ stick].]$$

John is the agent for the *seeing* event e_1, while Bill, Mary and stick are the agent, theme and instrument respectively for the *hitting* event e_2. Thus, semantic roles give a systematic method for capturing the event structure of a sentence. They also help in representing the semantic information contained within a sentence, since the value that a role takes is independent of the syntactic structure of the sentence.

This paper presents a corpus-based approach for obtaining a semantic role for different constituents of a sentence based on their syntactic roles with respect to a target constituent of the same sentence. This approach is an improvement over [3]. Since the approach is entirely statistical, we expect that it is language-independent and thus can be used for similar processing of Indian language texts, as discussed in Section V.

A. Syntax to Semantics

Our approach is based on *linking theory* [5], which claims that there is a unique relationship between the syntactic and semantic structure of a sentence. Thus the syntactic structure can be used to determine the semantic role for the various constituents of the sentence. The specific approach follows [3] and uses a set of parse tree features:

1) **Phrase Category (h)**: Different roles tend to be realized by different syntactic categories. This feature indicates the syntactic category of the phrase expressing the semantic roles.
2) **Governing Category (g)**: This feature, defined only for NPs, has only two values, S and VP, corresponding to subjects and object of the the target verb, respectively. Since subject and object often tend to take different semantic role in a sentence, this feature is important to correntcly identify the semantic role associated with a constituent.
3) **Parse Tree Path (pt)**: This feature is designed to capture the syntactic relation of a constituent to the rest of the sentence. The value of this feature is the string constructed from the traversal of the parse tree from the constituent to the target verb.
4) **Position (pos)**: This feature simply indicates whether the constituent to be labeled occurs before or after the the target verb in the sentence. This feature is used to ensure that even if the rest features are not accurately determined by parsing error, still some minimal information could be obtained so as to predict semantic role for the constituent.
5) **Head Word (t)**: Head words of NPs can be used to express selectional restrictions on the semantic types of role fillers. Hence, this feature, which is available from the parser output, is used.

In order to illustrate these features better, consider the parse tree of the sentence *He heard the sound of water slurping in a metal container as Farrell approached him from behind* shown in Figure 1.

Now, considering *approached* as the target, the values of these features for the constituent *Farrell* will be as follows:

1) Phrase Category: NNP
2) Governing Category: S
3) Parse Tree Path: $VBD \uparrow VP \uparrow S \downarrow NP \downarrow NNP$
4) Position: $Left$
5) Head Word: $Farrell$

It can be observed that these features form a rich quantification of the syntactic structure and hence are appropriate for our purpose. We next explain how these features can be used to predict constituents and roles in a sentence.

II. A Probabilistic Model for Semantic Roles

As was emphasised in the previous section, the syntactic features of a constituent influence the role it takes in a sentence. However, in order to predict this role on the basis of these features we need a mathematical model of the dependence between the roles and syntactic features. An evident choice for this is a probabilistic model, which may be

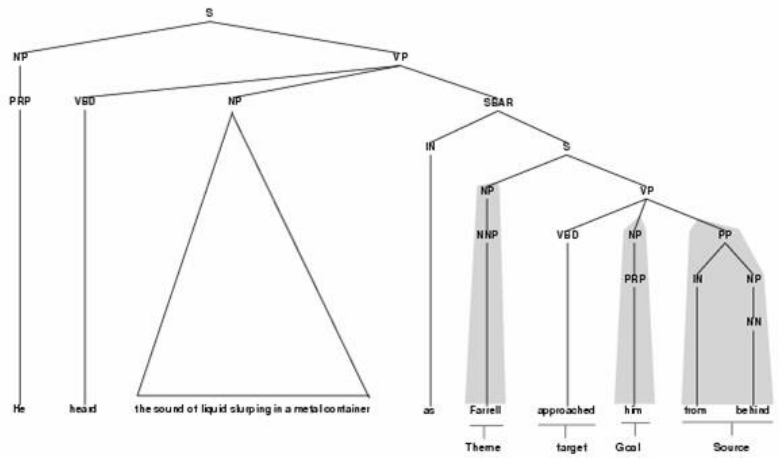


Fig. 1. Example parse tree

S.No.	Probability	Conditioning Variable
1	$P(c\|pt)$	Parse-tree path
2	$P(c\|t,pt)$	Parse-tree path and head word
3	$P(c\|t,h)$	Head word and phrase type
4	$P(c\|pt,h)$	Parse-tree path and phrase type
5	$P(c\|h)$	Phrase type
6	$P(c\|g)$	Governing category

TABLE I
CONDITIONAL PROBABILITIES USED FOR CANDIDATE IDENTIFICATION

specified as follows:

$$P(r|h,g,pt,pos,t) = \frac{n(r,h,g,pt,pos,t)}{n(h,g,pt,pos,t)} \quad (1)$$

where the quantities $n(r,h,g,pt,pos,t)$ and $n(h,g,pt,pos,t)$ are estimated from the training data. However, an important point to note here is that the linguistic data used for this estimation is very diverse, and hence the frequency of occurence of the tuples (r,h,g,pt,pos,t) and (h,g,pt,pos,t) may not be sufficient to estimate the conditional probability. So, in order to ensure the usefulness of the model, we break the original probability into an average of simpler conditional probabilities like $P(r|h,pt,t), P(r|h,t)$ etc.

On the basis of the above reasoning, our approach can be broken down into two stages, namely **constituent identification** and **role assignment**. Both these stages are based on a set of conditional probabilities which are combined to estimate the goodness of a selection.

A. Constituent Identification

This stage comprises of assigning probabilities during the training phase to various conditioning variables which may indicate if a sentence constituent is a candidate for role assignment, and then using these probabilities during the testing phase for identifying the most probable candidate in a sentence input for semantic tagging.

The conditional probabilities used for identification of role candidates are tabulated in Table I. Only the first three probabilities have been considered by [3], and these indicate a greater emphasis on the headword as a feature. However, we observed that tree-path and phrase category are more informative features about the syntax and hence added the last three probabilities to make the approach intuitively stronger. The results were found to be improved by this inclusion.

The constituents identified from this part are passed to the role assignment stage.

B. Role Assignment

In this part, during the training phase probabilities are assigned to conditional variables which influence role assignment, and these probabilities are used in the testing phase to assign roles to the constituents identified in the previous part.

The conditional probabilities used for identification of roles are tabulated in Table II. Again, only the first eight probabilities have been used by [3]. Also, all these probabilities carry equal weightage in the simplest version of their algorithm. We included features based on tree-path and phrase type here also, and assigned weights to the final set of probabilities. The use of weights, which was largely done after repeated experiments, enabled us to assign greater importance to the conditional probabilities which we expected to influence the role assignment significant. For instance, the conditioning variables in $P(r|h,pt,t)$ are Phrase Type, Parse Tree Path and Head Word. Of these, the first two, in our experience, were the strongest features for role assignment. Similarly, Head Word was crucial since if the same word appears in a test sentence, it is highly likely that it will take the same role as that in the training sentence in which it appeared. We found that the inclusion of weights yielded better results than the original algorithm.

At the end of these two stages, we get the most probable roles for the most appropriate constituents of the input sentence. The performance of this algorithm is discussed next.

III. DATA DESCRIPTION

Our system makes use of two other systems. The first is the FrameNet database [2], which contains role annotations

S.No.	Probability	Conditioning Variable	Weight
1	$P(r\|t)$	Head word	1
2	$P(r\|pt,t)$	Parse tree path and head word	1
3	$P(r\|pt,g,t)$	Parse tree path, governing category and head word	1
4	$P(r\|pt,pos)$	Parse tree path and position	2
5	$P(r\|pt,pos,t)$	Parse tree path, position and head word	1
6	$P(r\|h)$	Phrase type	4
7	$P(r\|h,t)$	Phrase type and head word	1
8	$P(r\|h,pt,t)$	Phrase type, parse tree path and head word	8
9	$P(r\|pt)$	Parse tree path	4
10	$P(r\|pt,h)$	Parse tree path and phrase type	2

TABLE II
CONDITIONAL PROBABILITIES USED FOR ROLE ASSIGNMENT

of sentences for various target verbs. The other is the Collins parser [6], which is a statistical head-driven parser and works on similar tags as the FrameNet database.

A. FrameNet

The FrameNet database [2] consists of several semantic frames containing words which tend to take a fixed set of semantic roles in normal English usage. Each of these frames further contains semantically annotated sentences, and defines a set of roles that can appear in the context of use of a particular word (sense) and assigns these roles to the appropriate part of the sentence. In addition, each word in each sentence is syntactically tagged using the tag set prescribed by the Penn Treebank Project [8]. This makes it compatible with the Collin's parser [6], which we discuss next.

B. Collin's Parser

Collin's Parser is a statistical natural language parser [6]. It is based on head-driven statistical models for natural language parsing. The parser has been trained on the Penn Treebank database [8] and takes as input a tagged sentence. The output from the parser not only consists of a parse tree but each non-terminal is also associated with a unique head-word.

IV. EXPERIMENTS AND RESULTS

For the purpose of training and testing, three frames from FrameNet were chosen:

1) **Experiencer_subj**: The words in this frame describe an experiencer's emotions with respect to some content. A Reason for the emotion may also be expressed. Example lexical units in this frame are *love, hate, dislike, mourn, want* etc. Also, an example of the annotated sentences in this frame is *His parents DISPAIRED of him because of his smoking habits*, where *his parents* is the Experiencer, *him* is the Content and *his smoking habits* is the Reason with respect the verb *dispair*.
2) **Conversation**: The words in this frame describe situations in which two (or more) people (the Interlocutors) talk to one another. Example lexical units in this frame are *discuss, converse, yak* etc. A sample annotated sentence from this frame is *They had a DISCUSSION about the scandal*, where *the scandal* is the Topic with respect to *discuss*.
3) **Experiencer_bodily_harm**: In this frame, An experiencer is involved in some event which results in bodily injury to a body part. Sample lexical units in this frame are *hit, cut, hurt* etc. An example annotated sentence from this frame is *He SMACKED his head on the mantel*, where *He* is the Experiencer, *his head* is the Body_part and *the mantel* is the Injuring_entity.

The annotated sentences for the verbs in each of these frames, approximately 1300, 900 and 600 in number respectively for the Experiencer_subj, Conversation and Experience_bodily_harm frames, were broken in an 80-20 ratio for training and testing. The average length of sentences in these frame was about nine words. Prior to both training and testing, these sentences were parsed using the Collin's parser and fed into the system.

Table III presents a summary of the accuracy obtained from the system with respect to its various parts. From the figures, it is evident that our algorithm was able to identify a significant percentage of role-value pairs correctly in the data, particularly considering the complexity of the sentences considered. Notably, the performance of our system was better than that reported in [3], though the tests performed in the latter were more exhaustive. Also, it must also be mentioned that in some cases, our algorithm fails to identify the appropriate roles and/or constituents, as in the sentence

Mayer knocked the director down to the floor and threw him out of his office.

for which the roles assigned are as follows:

- *Body_part*: to the floor
- *Victim*: the director
- *Agent*: Mayer

Here, while the last two assignments are accurate, the constituent *to the floor* has been assigned an incorrect label. This clearly shows that the present system should be trained over more diverse data and should be further tuned to yield more accurate results. This tuning could be in terms of the use of more conditional probabilities for constituent identification and role assignment, optimal assignment of weights and betters method of synthesizing simple conditional probabilities into complex ones.

Frame→	Experiencer_Subj	Experience_bodily_harm	Conversation
Constituent identification accuracy	69.48	73.63	63.96
Role assignment accuracy for correctly identified constituents	74.90	66.98	67.60
Overall role assignment accuracy	52.04	49.31	43.24

TABLE III
PERFORMANCE SUMMARY

V. DISCUSSION

In this paper, we presented a strategy for semantic role tagging using statistical methods. The approach is based on linking theory, which is found to hold for Indian languages as well. Beyond the extraction of the features, the algorithm is purely statistical and is thus unbiased towards any particular language. Also, the set of features considered is quite complete and the way these are composed into conditional probabilities is language independent. Last but not the least, synonymous lexical units in different languages are expected to play the same role in similar sentences. Hence, we expect this approach to work well for Indian languages also. However, this requires considerable work on creating a semantically tagged corpus as in FrameNet. This work is currently underway [7].

REFERENCES

[1] Donald Davidson: “The Logical Form of Action Sentences”: The Logic of Decision and Action, University of Pittsburgh Press, pp 81-95: 1967.

[2] Collin F. Baker, Charles J. Fillmore and John B. Lowe: ”The Berkeley FrameNet Project”: Proc. the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and the Seventeenth International Conference on Computational Linguistics, pp 86-90: 1998

[3] Daniel Gildea and Daniel Jurafsky: ”Automatic Labeling of Semantic Roles”: Computational Linguistics, Volume 28, Number 3, pp 245-288: 2002.

[4] Kadri Hacioglu, Sameer Pradhan, WayneWard, James H. Martin and Dan Jurafsky: ”Shallow Semantic Parsing using Support Vector Machines”: Technical Report, TR-CSLR-2003-03, Center for Spoken Language Research, University of Colorado: Oct 2003.

[5] David Dowty: ”Thematic Proto-Roles and Argument Selection”: Language, Volume 67, pp 547-619: 1991.

[6] Michael Collins: ”A New Statistical Parser Based on Bigram Lexical Dependencies”: Proc. the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics, pp 184-191: 1996 (http://www.ai.mit.edu/people/mcollins/code.html).

[7] Pankaj Goyal, Ankit Soni, Deepak Sharma, Amitabha Mukerjee and Achla Raina: ”A Frame-Semantic Approach for Tagging Hindi and Bangla Sentences”: Syposium on Indian Morphology, Phonology and Language Engineering: 2004

[8] Mitchell P. Marcus, Beatrice Santorini and Mary A. Marcinkiewicz: ”Building a large annotated corpus of English: the Penn Treebank”: Computational Linguistics, Volume 19, Number 2, pp 313-330: 1994

Parts of Speech Tagger for Tamil

Arulmozhi. P, Sobha. L, Kumara Shanmugam. B.
AU-KBC Research Centre, MIT campus of Anna university,
Chennai – 44.
E-mail: (a_mozhi,sobha,bkumar)@au-kbc.org.

1. INTRODUCTION

The parts of speech (POS) tagging is an essential task for all the language processing activities. The POS tagger of any language assigns an appropriate part of speech tag for each word in a sentence. Essentially this is done after morphological analysis. The morphological analyzer may give multiple output for a word and the ambiguity is resolved by the POS tagger.

For example,

Take the book. – VB DT NN
Book the ticket. – VB DT NN

In both the sentences, book plays different roles – Noun in the first and Verb in the second. The morphological analyzer may give both the tags (noun and verb) for the word 'book'. The POS tagger solves the ambiguity and gives the correct/single tag for that word.

2. POS TAGGING METHODS

There are various methods in which POS taggers are designed. The most common methods are

1. Stochastic Taggers
2. Rule Based Taggers
3. Transformation Based Learning method

2.1. Stochastic Taggers:

Stochastic taggers generally resolve the ambiguity by computing the probability of a given word (or the tag). The probability is calculated using a training corpus. The training corpus is a tagged corpus, which is assumed 100% correct. The probabilities are calculated using unigram, bigram, trigram, and n-gram methods.

2.2. Rule Based Taggers:

These taggers are based on a defined set of hand written rules. Most of the existing Rule Based POS taggers are based on two-stage architecture. The first stage assigns a list of probable tags (or the basic tag) for a particular word. The second stage, uses the list of disambiguation rules, to reduce the list (or change a wrong tag) to a single right tag.

Here all the rules are pre-defined. They may be language dependent or independent. The rules can be broadly classified in to two.

1. Lexical Rules
2. Context Sensitive Rules

The lexical rules act in a word level and The context sensitive rules act in a sentence level.

2.3. Transformation Based Tagging:

This is an approach in which the rule-based and the stochastic methods are combined and used. Like the rule based taggers, this is based on rules, which say what tag to be assigned to what words. And like the stochastic taggers, this is a (supervised) machine learning technique, in which the rules are automatically derived from the (pre tagged) training data. And those rules are applied to the test corpus.

There are various algorithms, which are used for learning purpose. The most commonly used Brill's tagger (for English) is a TBL based tagger. Brill's algorithm has three major stages. In first stage, it labels every word with the most likely tag. In second stage, it examines all the possible transformations and selects the one, which gives the most improved tagging.

3. POS TAGGER FOR TAMIL:

Tamil is a morphologically rich language. The gramatical relations (tense, gender, number,, person etc.) are expressed by the suffixes which are added to the words. Some of the suffixes when added with a root word, the root word can take the POS according to the suffix. So, in the lexical level, the suffixes play a major part in deciding the POS of a word.

A POS tagger which finds out the major POS (N, V, etc) of the root word is developed using rule-based method. This tries to find the POS of the root word using the inflection of the word without using any root word dictionary. Stochastic taggers like HMM will not give a high accuracy for Tamil because the language is inflectionally rich and is relatively free-word order.

For Example,

The word **"patikkira:n"** can be splitted in to

pati	**+**	**kkir**	**+**	**a:n**
verb	+	Pres	+	Sng.M.3P

The root word "pati" cannot be identified without a dictionary. But the tense, Gender, Number, Person, plural markers can be identified with a limited list of suffixes. There are suffixes for verb and noun as well as suffixes which can only occur with adjectives and adverbs.

In the above example, using the tense marker and GNP marker, the root word can be assumed as a verb. If there is any ambiguity, the word is left as unknown and it is resolved in the next step. The supporting tables are Suffix list, Lexical rules and context sensitive rules.

The suffix list is a table in which the suffixes and its appropriate tables are written. For example,

"-ai" suffix for accusative marker is represented by **"ai 1*1**" – means, the suffix **"ne"** belongs to the first sub table of the first table.

Al,1*2 – means, the suffix **"Al"** belongs to the second sub table of the first table. Totally there are 20 tables (excluding the sub tables).

The lexical rules act in a word level, for example,

Lexical rule: N*ACC,1*1

Example:

Penavai "Pen"	penavai
ai - 1*1	N*ACC

The suffix "ai" is an accusative marker, which can occur only with the noun.

While stripping off the suffixes, the process is not done till the end of the word. It is stopped when minimum of two characters is left in the word. This is done because for a root word, this is the minimum length. This is one of the heuristics used in the system. Context sensitive rules act in the sentence level. For example,

In the sentence,

"ravi oru ma:mpalyam ca:ppitta:n."

Ravi one mango eat+pst+m+sng+3p

(Ravi ate a mango)

The lexical tagging will be,

"ravi <V> oru<Q> ma:mpalyam<N> ca:ppitta:n<V>"

Here, ravi is tagged as a verb because of its 'i' ending which is actually a proper noun. Since Tamil is a verb final language a verb cannot be in the initail position of the sentence. This is solved by the context sensitive rule. So the **context sensitive rule,**

{ -1, 0,V } 0,V,N;

which means – if there is no previous word and the current word's tag is V, (that is if the starting word is tagged as V) then change the current tag from V to N. There are 6 context sensitive rules and several heuristic rules applied in this system.

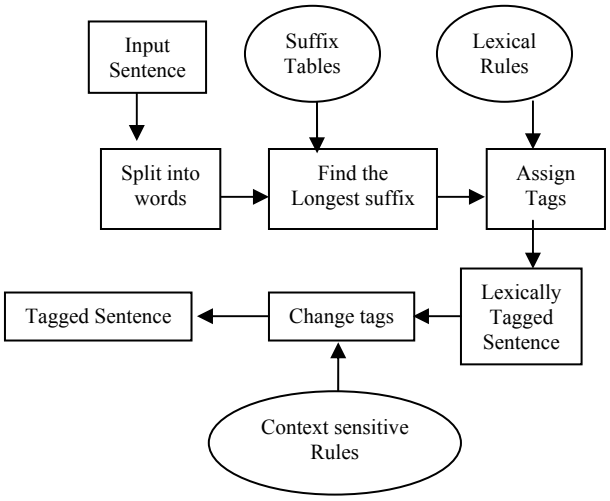
One of the heuristics made here is to assume the most likely tag for the words, which are left unknown after applying the context sensitive rules. They are assumed as nouns. The reason for that is they are most probably proper nouns or foreign words.

4. ALGORITHM

The suffix stripper uses a list of suffixes, pronouns, adjectives and adverbs. The input format is one sentence per line in which each word is separated by a white space. On the input text, it performs the following steps:

1. Split the sentence in to words.
2. For each word,
   - 2.1. find the longest suffix at the end
   - 2.2. find the table number of the suffix and eliminate the suffix from the word
   - 2.3. Go to 2.1 until the word length is 2.
3. Using the combination of suffixes and the rules, apply the lexical rules and assign the category.
4. For each sentence,
   - 4.1. Apply the context sensitive rules on the unknown words.
   - 4.2. Apply the context sensitive rules on the wrongly tagged words.
   - 4.3. If no context rule applies for any unknown words, tag it as noun.

The block diagram of the suffix stripper is shown below.



Block Diagram for Suffix stripper

5. TAGSET

A tagset is the set of Part of speech categories, in which, any word in a language can fall in to any one of those categories. And it gives the representation for each of the POS tag. There are various tagsets used for tagging a English corpus. Some famous tagsets are PennTreebank tagset, C5 Tagset, C7 Tagset etc. The tagset for suffix stripper contains 12 major categories. They are,

1. N - Noun
2. V - Verb
3. J - Adjective
4. A - Adverb
5. Q - Quantifier
6. C - Conjunction
7. P - postposition
8. PRO - Pronoun
9. QUES - Question word
10. VBN - VerbalNoun
11. SYM - Symbol
12. NUM - Number

6. EVALUATION:

The system performance is evaluated against CIIL corrected corpus (3 million). Two sets of thousand words each were taken and manually evaluated. The system gives 92% correct tags for each word.

$$\text{Precision} = \frac{\text{No. of correctly tagged words}}{\text{No. of total words}}$$

The sentences were taken randomly from the CIIL corpus and evaluated. Training fn-TBL with hand annotated corpus was not very effective because of the inflections in the root word and it tried to give the sub-tags also for that. The evaluation table is given below.

No. of Tested Words	Totally Tagged words	Correctly Tagged Words	Precision
1000	1000	920	92%
1000	1000	918	91.8%

The evaluation was done in two stages. Both after applying the lexical rules and after applying the context sensitive rules.

After the Lexical rules were applied on the list of words, the precision with the subtags was 83.66% and when we don't consider the correctness of the sub tags, the precision was 89.04%

The context sensitive rules were applied on the output and the evaluation is done. The precision was 88.6 when the correctness of the sub tags was considered and it was 93.22% when the sub tags are overlooked.

7. CONCLUSION

The work shows a simple rule-based Parts of Speech tagger for a morphologically rich language. The tagger gives correct tags for all the inflected words, which takes the precision to a higher level without any root word dictionary or training corpus because the un-inflected words are very less and many of them are handled by context sensitive rules. This tagger can be trained in future for generating the rules automatically using machine learning algorithms and a tagged training corpus.

REFERENCE

[1] D. Jurafsky & J. H. Martin. (2000) *Speech and Language Processing*. Parson Education

[2] E. Charniak (1997). *Statistical Language Learning*. MIT Press, Cambridge, London.

[3] C. D. Manning and H. Schütze. (1999) *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge

[4] E. Brill (1992) *A Simple Rule-Based Parts of Speech Tagger*, University of Pennsylvania.

[5] B. Megyesi, (1999) *Improving Brill'S POS Tagger for an Agglutinative Language*, Stockholm University.

Morpheme and Parts-of-Speech tagging of Tamil corpus

M Ganesan **S. Raja**
CAS in Linguistics
Annamalai University

1. Introduction

Language corpus is a collection of texts of the written and spoken words, which is stored, in an organized way, in electronic media for the purpose of linguistic research. It serves as a resource to be systematically 'consulted' by language investigators. The potentiality of large corpora (in English and other European Languages) has been recognized by speech and information technologists besides linguists and lexicographers. The corpora are further analysed and the linguistic information are enclosed at different levels (tagged corpora) for the convenience of researchers to retrieve selective information automatically. The main areas where corpora are found to be useful are linguistics, lexicography, natural language processing, language teaching and speech processing.

For the first time the texts of Indian languages are made available in machine readable form through the project 'Development of Corpora of text of Indian Languages' started in1991 by Department of Electronics (DoE), Govt. of India. The corpora development project for the 15 scheduled languages has been undertaken by six different centers. Later languages, newly added to the 8th schedule have also been included for building corpus. The objective, size of the corpora, coordination between centers, etc. have been discussed elaborately by Annamalai (1994). The Central Institute of Indian Languages, Mysore has taken up the corpora development work for Kannada, Malayalam, Tamil and Telugu Languages. This paper explains an approach for tagging the corpora automatically at word and morphemic levels for Tamil. It also gives different tag sets used at both the levels.

2. Corpus Management

A corpus can hold a number of information besides the texts which, in turn, makes the information retrieval a relatively trivial task. These information are broadly grouped into two types: 1) representative information (actual form of the text) and 2) interpretative information (adding linguistic information to the text)(Leech, 1993: 275). In order to store the text in a structured way, software called CATT (Corpus Analysis Tools - Tamil) developed by Ganesan, one of the present authors is used. The representative information that are stored within the body of the corpus are major and sub categories of texts, source, date of origin, authorship and publishers. Using CATT one can also retrieve these texts selectively. For example, one can extract all the texts grouped under a particular sub category or the texts from a particular period, etc.

3. Corpus Annotation

Corpus can be made to provide more useful information even on individual sentences, words, morphemes, etc. It could be achieved by adding linguistic information (interpretative information) to the text. The practice of adding linguistic information to an existing corpus of spoken or written language by some kind of coding attached to, or interspersed with, the electronic representation of language material itself is called corpus annotation (Ibid), such annotations can be made at different levels, namely, orthographic, phonetic/phonemic, prosodic, grammatical, syntactic, semantic and pragmatic/discourse level (see Leech, 1993). One of the basic advantages of annotated corpora is that the structural information at different levels could be retrieved based on linguistic tags, which are the frequent requirements of linguists, lexicographers and NLP researchers.

4. Grammatical Tagging

Grammatical tagging is the popular and common type of annotation successfully implemented in a number of corpora in English and other European languages. It is the procedure, which adds a tag at the end of a word to indicate its grammatical category. It can be achieved in two ways: 1) manual tagging and 2) automatic tagging (with manual post-editing). The

former is slow, labour intensive and liable to error and inconsistency (Leech, 1992; 131). There are different approaches in the latter, but can be broadly grouped into two methods: 1) rule – based tagging and 2) statistics – based tagging. The former method is explained in the following sections.

6. Tag- sets

The corpora of Indian Languages were tagged with a bare minimum of word-tags, i.e., only with 7 word-tags. They are 1) noun (NN), 2) pronoun (PN), 3) finite verb (FV), 4) non-finite verb (NV), 5) adjective (AJ), 6) adverb (AV) and 7) indeclinables (ID). As the corpora envisage multiple uses, it was decided to limit the tagging only to the major seven parts of speeches. Currently Tamil corpus has been tagged with more number of tag sets at word level and an elaborated labeling at morpheme level are carried out in order to meet the requirements different user group. There are 34 tags at word level and 132 tags at morpheme level.

6. Problems pertaining to tagging of Tamil Corpus

i) ***Identification of words***: Normally a sequence of characters between two successive spaces is considered as a word. It is even convenient to the computers to identify a unit as a word. But in real sense, the unit need not always be a simple word, i.e., it may be a compound or conjoined word, where the base form does not find a place in the dictionary.

ii) ***Internal sandhi***: The morphophonemic changes that take place when a suffix is added to a stem depend on the final phoneme of the stem and the initial phoneme of the suffix and which are too many in the agglutinative languages.

iii) ***External sandhi***: The operation of external (the morphophonemic change that takes place when two words are conjoined) is not consistant in some languages like Tamil.

iv) ***Inconsistancy in spacing between words***: In Tamil two or more independent words are written jointly as a single unit. Sometimes inconsistency persists in spacing between main and auxillary verb, noun and particle, etc.

7. Tagging scheme

The approach for grammatical tagging adopted is mainly based on the morphological analysis of these languages. To segment a word (if it has more than one morph) to its stem and suffix (es), the word can be approached either from the beginning (Left to Right) or from the end (Right to Left). The scheme, which we follow, approaches the word from the end in order to detach the suffix (es) one by one from the stem, as suffixes are finite in any natural language. The system first identifies the valid morph in the word one by one and label them at morpheme level then the entire word is tagged for its grammatical category at word level. This system has three major components: 1) Stem- MRD (Machine Readable Dictionary), 2) Suffix – MRDs and 3) a set of morphophonemic rules.

7.1 Stem – MRD

The major tasks involved in the preparation of stem- MRD are the collection of words, identification of their stem alternants and classification. Stem-MRD consists of all the possible roots and stems in the language. For example, if a word has four stem alternants, all the four stem will be included in the dictionary as independent entries. They are classified into various types on the basis of the first suffix they take. The basic structure of the stem-MRD is as follows:

Stem / Category / Type / Status

7.2 Suffix – MRDs

The basic principles underlying in the design of different MRDs for suffixes are the position of a suffix in a word and its companion. In our system the searching begins from the end of a word. The system identifies and detaches the suffix (es) one by one till it finds a stem. It is performed using a number of suffix-MRDs rather than one. The basic structure of the suffix – MRD is as follows:

Suffix / Type / Morpheme-tag / Word-tag.

The suffix – MRD also consists of four fields. The actual suffix occupies the first field. The number in the second field indicates the type of suffixes, which could occupy the immediate left position of the present suffix. It actually helps to select the proper MRD for searching. The third field gives the grammatical information of the suffix which would be used to tag at the morphemic level. The last field indicates word-tag information, if this suffix is

the determining element. The last two fields may contain more than one entry, when the suffix has different grammatical functions in different contexts. As the order of suffixation is unique for any word form, it would be easy to condition the occurrence of a given suffix. So the type number that explains this condition plays a crucial role in the analysis.

S_1, S_2, etc. given in the type-field indicate that the possible previous element would only be a stem and that stem belongs to a particular group. The information on the stem group is made available in S-file. The S-file for the above example is as follows:

7.3 S-file

$S_1 > 1,2$
$S_2 > 2$

If the suffix indicates the type as S_1, then the possible stems are of type 1 and 2(type given in the stem – MRD) only.

7.4 Morphophonemic Rules

The third component of the system is a set of morphophonemic rules, which operate externally. It is necessary for reverting the sandhi operation in order to obtain the stem and suffixes of the word encountered, as given in the MRDs.

7. Brief Description on the operation of the System

The system reads a word from the corpus and tries to match with those entries marked ID as status in the stem – MRD. If it succeeds, it records the category found in the second field and tags the word appropriately. If fails, it tries to segment the last suffix and to match with suffixes, listed in L_1 – MRD. If it finds a match, based on the value in the type–field it proceeds to the respective suffix–MRD or the stem–MRD. This procedure is continued till a stem is reached. At each success point the grammatical value and the word–tag information (if any) is stored along with the morpheme and the position. When the word encountered is completely analysed, the stored information as required will be written into the output file. If the system does not find a match in the last element itself, it tries to use the morphophonemic rules to revert the sandhi operation. If there is any success, the system repeats the procedure from the beginning. In case of failure, the word will be left untagged and the next word will be taken for analysis. Similarly, and disagreement in the matching at any stage beyond L1 leaves the word untagged.

As all the alternant forms of stems and suffixes are included in the MRDs, the problem of the internal sandhi is easily solved. In this model, when the system encounters more than one grammatical category for a suffix, it first attempts to analyse the whole word for the first category and then restarts the analysis for the second category, and so on. So the system is capable of analysing the homophonous forms for all their possible structures.

This model also resolves the problems of compound and conjoined words which are found with or without space to a maximum extent. Most commonly used compound forms are included in the stem – MRD. The other compound and conjoined words are tackled using repeated procedure i.e., every time after finding a stem, the system looks for any remainder. If there is, it repeats the analysis from the very beginning, as if the remainder is a new word. The untagged words and the words with more than one tag can be manually tagged.

9. Conclusion

The system discussed in this paper is mainly evolved to handle the languages, which are morphologically rich. The system is language independent. This model with some drop-outs of procedures can be used for spell- checker while considering the three desiderata, speed, consistency, accuracy indicated by Leech (1993:279) for a tagging scheme, this system may be slow. But the speed, from my point of view, need not be considered on par with the other two criteria as tagging as corpus is a one on time job. However the speed of application can be considerably in this system, by building a single suffix–MRD Based on the position alone (ignoring the companion) in which case the corpora should be free from spelling and grammatical errors.

CORPORA & LEXICAL RESOURCES

Annotated Speech Corpora Creation: An Approach

Shyamal Kr. Das Mandal
Centre for Development of advanced Computing (C-DAC), Kolkata
shyamal.dasmandal@erdcical.org

Asoke Kumar Datta
Indian Statistical Institute, Kolkata

Abstract

For any kind of research work the standard sample data collection is crucial. For the development of speech technology and speech research the standard unnoted speech data (corpora) plays a very important role. The speech corpora provide the important voice segment to the researcher for making test bade, manipulation, analysis, and collect statistics of the different speech parameter. Good speech corpora should content the basic element of speech research like acoustic phonetics and acoustic prosodic and also the required data for speech technology development. Various speech corpora by different groups have been made, but all of these are directly created & managed by the core group. In this paper we have discusses about an approach for building and management of annotated speech corpora guided by the needs of Acoustic Phonetics, Acoustic Prosodic and the large number of speaker variation for speech recognition.

1. INTRODUCTION:

In this age of information technology, information access in a convenient manner has gained importance. In the current Indian context, the machine-oriented interfaces restrict the usage to a miniscule fraction of the population, which are both computer literate and conversant with written English. Since speech is a primary mode of communication among human beings, it would be convenient to have a human computer dialogue in audio mode to take place, if possible in local languages. This demands research and development in the areas of speech recognition, natural language processing and speech synthesis. Since India is multilingual country so there is a tremendous potential research required for the development of Speech technology in different languages to benefit the masses for getting advantage of Information Technology. For the development of Speech Technology and Acoustic Phonetics Research of different language demands collection of annotated Speech Corpora. Since annotated speech corpora have been a critical component of research in the speech sciences. Various speech corpora by different groups have been made, but all of these are directly created & managed by the core group. The major issues involved are the propagation of repairs, consistency of references, and the ability to integrate annotations having different formats and levels of detail. Speech corpora should content the different type of speech segment with proper annotation and languages marking.

2. METHODOLOGY:

The corpora creation methodology can be divided in four parts: 1. Content Selection, 2. Recording 3. Tagging 4. Organization

2.1 Content Selection:

Content of the Speech Corpora can be divided into three parts according to the need:

1) Speech Research
2) Automatic Speech Recognition
3) Speaker Recognition

Figure.1 illustrates how these needs help to generate the contents of speech corpora. The target content is given within rectangular boxes. The intermediate flow paths indicate the interaction of different technological and research needs.

2.1.1 Speech Research:

In the area of speech research there are two aspects one is acoustic phonetic and other is acoustic prosodic.

a) Acoustic phonetics:

As the acoustic properties of a phoneme depend significantly on the adjoining phonemes there is a need to have phonemes sound in all possible bi-gram contexts. Furthermore, these also depend on speaker.

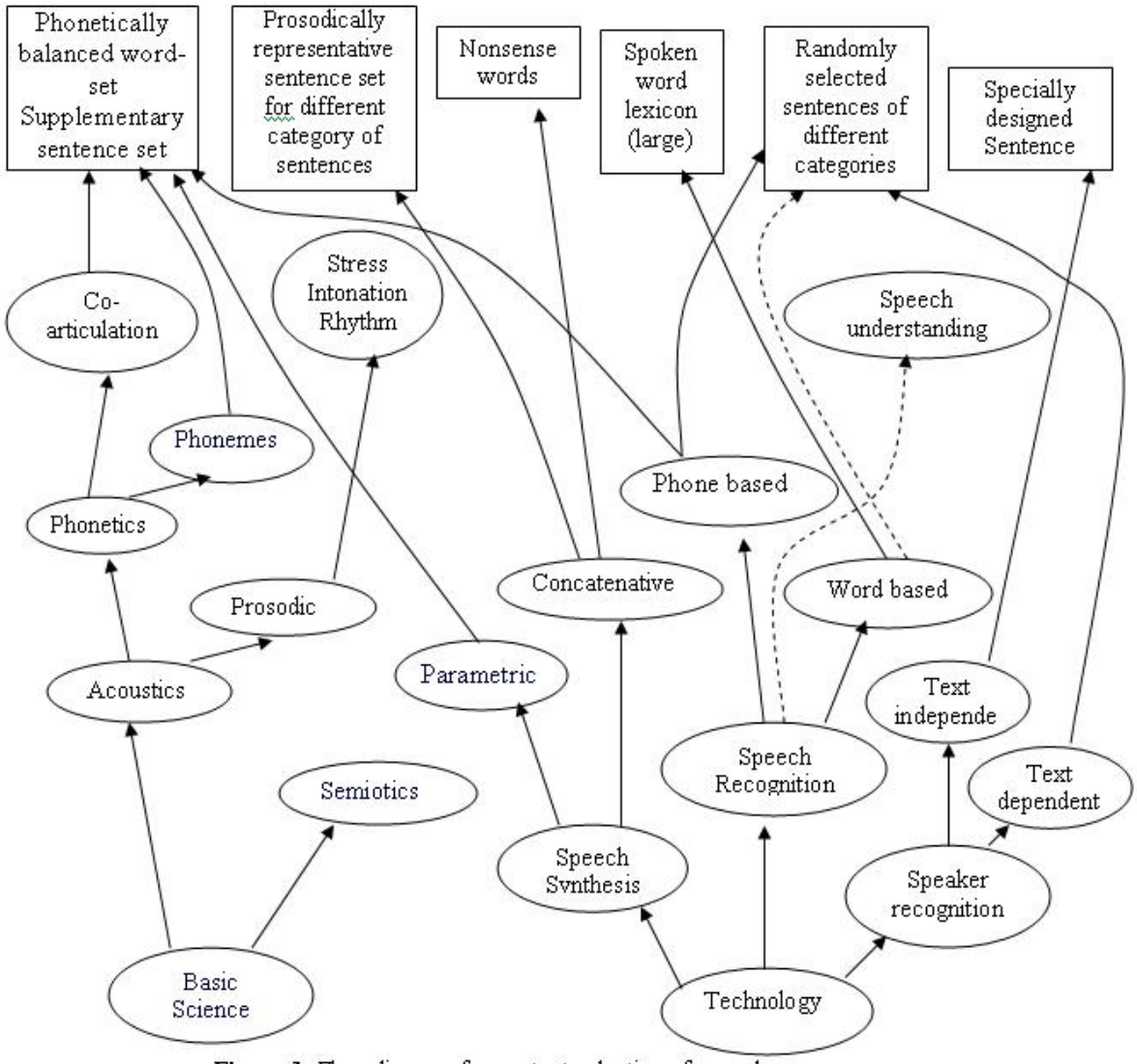


Figure 1. Flow diagram for content selection of speech corpora

It is therefore necessary to take the phonologically balanced word set for context variation to be spoken by different speaker a number of times to include speaker variation.

b) Acoustic prosodic

Prosodic pattern, though have a common characteristics, depends strongly on the types of sentences, mode of speaking as well as the individuality of a speaker. Therefore, it is necessary to record all the different type of sentences for a particular language by the different speaker in different mood. For the above requirement the text should be selected carefully by an experts linguists so that it cover the most of the prosodic structure of the particular language.

2.1.2. **Automatic Speech Recognition**

The development of Automatic Speech Recognizer (ASR) for continuous normal speech may be the toughest task. Isolated word recognition is considered to be the initial problem; an intermediate stage is slow dictation mode where the word is distinctly spoken. The corpora should provide ample database for each of these tasks.

2.1.3. Speaker Recognition

Text dependent speaker verification system requires a speech database, which must be spoken by a large number of speakers. A small set of optimized sentences that is able to discriminate a large number of speakers is the first requirement. An important requirement of the speaker verification system is that it must be resistance to possible intrusion by mimic.

2.2. Recording

After the selection of all the content of speech corpora for a particular language it should be recorded by proper professional informant in studio environment. The recording standard of the whole corpora should be in 16-bit mono with 22050 Hz sampling rate PCM wave format recording at speech studio environment.

2.3. Corpora Tagging Mechanism

For the above corpora the tagging has to be done in phoneme label followed by syllable and word marker for the all word corpora. For the sentences corpora the sentence should be tagged in phone, syllable, word, phrase/clause and parts of speech.
The speech signal of the corresponding word or sentences should be stored in binary wave file format with proper name. Name is decided by the corpora management procedure. Each wave file has an associated tagged information file in same name where the tagged information is stored.
The phoneme markers to be placed on the wave file are kept as time value in the tag file. The tag files also contain special symbols corresponding to phoneme category, syllable, word, clause and parts of speech (figure 2.).

Tagging can be done using the existing speech analysis tools e.g. Cool Edit, EMU-Labeler. The text description of the tagged corpora is in Unicode XML format.

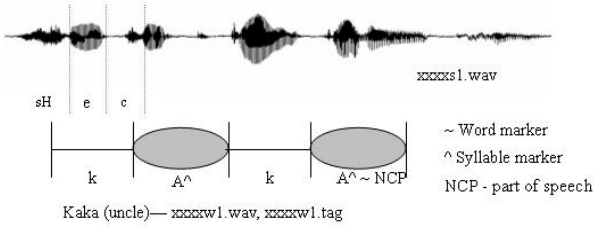


Figure 2

2.4. Organization:

The corpora have the two files corresponding each of the word or sentence for a particular field and each of the wave file and tagged file name is generated using three fields 1. Field ID, 2. Speaker ID, 3. Word or sentence ID as in figure 3. Wave information: the wave file for individual sentence/word is store in the from of XXYYZZ.wav and the tagged information file in the from of XXYYZZ.tag file format. XX=Field ID, YY=Speaker ID, ZZ=Word/Sentence ID

3. PHONETIC REPRESENTATION OF SPEECH LABELING

For phonetic tagging the following symbols are considered keeping in mind the need of a national

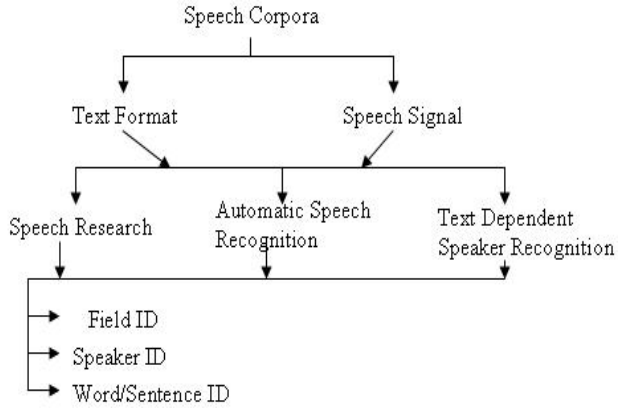


Figure 3

standard. In deciding this set of symbol ease of a common linguist and the availability of character and symbol set in the Windows Platform was taken into consideration.
In deciding the ASCII character set for phonetic transcription, tagging as well as writing the phonetic script corresponding to the grapheme text the following considerations are taken into account:
1) The corpora need to have national and international visibility. The IPA character is normally not available in PC and not common with data entry operators. The characters available with normal PC keyboards are very friendly to data entry operators. Furthermore these are c++ friendly and therefore handling and databases becomes simpler.
2) For ease of operation the additional attribute markers like markers for nasalisation, centralization, retroflexion etc are taken from the standard character set and placed next to the phoneme.
3) The representation of all the consonants of Indian languages require more than one character e.g., it necessary to distinguish between /k/ and /kh/, between /t/ and /th/ and the retroflexed versions of these. Use of two or more letters is necessary which again may cause ambiguity later in processing. It is therefore necessary to distinguish the additional letters. As the frequency of occurrence of the single letter consonants is high it is decided to use the small letters for the first letter and capital letters for

the additional ones. As for example k stands for /k/ and kH stands for /kh/ while h stands for the stand-alone /h/.

4) While some of the vowels in Indian spoken languages may have some small shifts in the position in the vowel diagram, in general, they may be represented by similar roman characters with some additional characters representing the special attributes as and when necessary.

For example representation symbol of speech sound for corresponding IPA Symbol for Bengali.

eE→ æ, k→ k, kH→ k^h etc.

Shorter counterparts of vowel in some language like Telugu are phonetically different. These are known to be centralized. These are indicated by a '-' next to the vowel e.g. :

e > e-

o > o-

Retroflexion is indicated by a '.' after the consonant symbol. Nasalisation of vowels is indicated by a '~'next to the vowel e.g., a > a~ etc.

4. REMARKS

C-DAC, Kolkata is building the speech corpora using the above approach for Standard Colloquial Bengali (SCB) and assumes and Manipuri. Our aim is to build such corpora where the speech scientist and the linguistic people can get the standard ready reference for speech research and speech technology development. We are initially building the corpora for SCB Bangla. The selection of the text for building the corpora is very important aspects. In case of acoustic prosodic part the selected sentences should content more or less all the prosodic pattern of the particular languages. Corpus is building for spoken languages so the selected text should in that from. We are taking the most recent story, novel, drama and the most popular local newspaper of Bengali for the collection of content.

The corpora cover only the standard spoken language not the dialectic variation of the language.

Acknowledgements

We thank to Prof. P. Sarkar and Prof. M. Mitra for their valuable guidance and suggestions for building the corpora and for this paper.

REFERENCES

[1] *TIMIT Speech Database*
http://www.stel.ru/speech/speech_database.htm

[2] *Russian Speech Database*
http//www.asel.udel.edu/icslp/cdrom/vol13/317/a317.pdf

[3] Dan T. K and Datta A. K. Speech synthesis: a time domain model. *Proc. Int. Workshop on recent trends in speech, music and allied signal processing*, 9-11 Dec, 1991, Delhi, India, pp. 39-45.

[4] Datta A. K, Ganguly N R, Mukherjee B. Intonation in segment-concatenated-speech. *Proc. ESCA Workshop on speech synthesis*, Sep 1990, France, pp. 153-156.

[5] *Development of Annotated Speech Corpora for 3 East Indian Languages (Bengali, Assamese, and Manipuri)* by C-DAC, Kolkata
http://www.erdcical.org/html/txttospeeh/intro.htm

Field	Area	Description	Speaker Identity	
			Male	Female
Speech Research	Acoustic Phonetic	450 phonologically balanced word set recorded with neutral carrier sentences	5	5
	Acoustic Prosodic	800 sentences of each of the following categories Text reading, Dialogue, Query-answer		
Speech Recognition	Isolated Word Recognition	Tagged most frequent words	10	10
	Slow dictation mode	Tagged sentence in slow dictation mode	10	10
	Continuous Normal Speech Recognition	Tagged sentence in normal speech reading	10	10
	Spoken Digit and number	Tagged different sequence in continuous and isolated. Digit and number	10	10
Speaker Recognition		Telephone bandwidth and normal specially designed sentences with some mimic with 5 times repetition	20	20

Table1: The corpora content

Linking Monolingual Resource with Bilingual Resource

Akshar Bharati **Vibhav Agarwal** **Sandipan Dandapat**
Rajeev Sangal **Dipti Misra Sharma**

International Institute of Information Technology Hyderabad
{vibhav,sandipan}@students.iiit.net, {sangal,dipti)@iiit.net

1. Introduction

This paper outlines the algorithm to link lexical resources available freely on the Net and Shabdaanjali, a free bilingual dictionary developed in-house at the Language Technology Research Centre, IIIT Hyderabad (LTRC). The algorithm also uses an application to combine the information available in Shabdaanjali and WordNet. WordNet gives different synset for a particular word and Shabdanjali has the senses of the word. The structuring of the hierarchy in which the senses appear in the dictionaries are different from the hierarchy of the synset appear in the WordNet. The challenge was to map the correct sense from Shabdaanjali to the closest synset from the synset list of WordNet.

2. The Problem

We used three sources of lexical information: WordNet, a semantic net of words arranged conceptually and programmed to offer output in terms of synonyms, antonyms, hyponyms, hypernyms, meronyms, etc.; a monolingual dictionary consisting of over a hundred thousand entries, an English-Hindi dictionary developed at LTRC. Each of the resources had their own strength and weaknesses, but the biggest problem stemmed from the fact that often, the number of senses allotted to words in each resource was different. For example, if Shabdaanjali listed only two senses of the noun Plant, WordNet listed four senses. How were we map the appropriate senses in each of this resource automatically to built more informative lexical resources for the future application?

3. Methodology

To link Shabdaanjali and WordNet, we have used three approaches:

(a) Dictionary Based Approach, measure the similarity between various Shabdaanjali senses of a given word with the WordNet synset of that word by using dictionaries.
(b) Corpus Based Approach, uses example sentences related to each sense given in Shabdaanjali to measure the similarity between sense and WordNet synset.
(c) Combined Approach, is a combination of previous two approaches, Dictionary-based approach and Corpus Based approach.

3.1 Dictionary Based Approach

The first approach to link Shabdaanjali and WordNet was based on sense similarity. It measures the similarity between various Shabdaanjali senses of a given word with the Wordnet synsets of that word by using dictionaries.

We used two dictionaries:
Shabdaanjali - An English-Hindi dictionary that consists of more than 27,000 entries.
Brahad Hindi Shabdakosha - A big Hindi-Hindi dictionary that consists of more than 1,00,000 Hindi entries.
The second dictionary Brahad Hindi Shabdakosha was used for enhancement of results.
For a given Shabdanjali sense sh_1, the similarity is calculated between sh_1 and a WordNet Synset by looking the senses of words of Synset in Shabdaanjali. If sense of the words of Synset and sh_1 is not matched then we go to Brahad Hindi Shabdakosha for further matching and a score is given. According to the score, mapping of sh_1 with a Synset was decided.

The words occurring in a Synset provide an essential clue towards the unique meaning conveyed by that synset. So, cluster of words of a synset was made to determine similarity between a sense of a given word with that cluster.
For example, “plant” has two senses in Shabdaanjali and four synsets in WordNet. The senses of “plant” in Shabdanjali are: “vanaspati” and “sanyantra”
And Synsets of “plant” in WordNet are
Synset 1: plant, works, industrial plant, building complex, complex
Synset 2: plant, flora, plant life, life form, organism, being, living thing

For a word 'w', do the following:

Store Shabdaanjali senses of 'w' in a list SHW

Make clusters of synsets of 'w' and store them in CLUSTER

For each entry i of SHW, do the following

Set $Score_{i, j}$ to '0'

For each cluster j of 'w', do the following

Match Shabdanjali senses of each word of $CLUSTER_j$ with SHW_i and if senses are matched then

$Score_{i, j} = 7 + (1/4)$

If some part of the senses is matched, give some penalty in the score

$Score_{i, j} = 7 + (1/4) * penalty$

If $Score_{i, j}$ is less than threshold value then set $Score_{i, j} = 0$

If $Score_{i, j} = 0$, Go to Hindi-Hindi dictionary for Shabdaanjali sense of each word of $CLUSTER_j$ and do the following

If senses are matched then

$Score_{i, j} = 4 + (1/4)$

If some part of the senses is matched, give some penalty in the score

$Score_{i, j} = 4 + (1/4) * penalty$

If $Score_{i, j}$ is less than threshold value then set $Score_{i, j} = 0$

Return $Score_{i, j}$

Synset 3: plant, contrivance, stratagem, dodge
Synset 4: plant, actor, histrion, player, thespian, role player

It is clear that cluster #1 is conveying sense of *vanaspati(botanical)* and cluster #2 is conveying sense of *sanyantra(industrial plant)*. Cluster #3 and cluster#4 are not supposed to match with any of two sense of plant given by Shabdaanjali.

The matching algorithm is given in above box.
For example, when the algorithm was tested on *plant*, we got following scores between two senses of *plant* and each Synset cluster.
Score(vanaspati, Synset#1) = 0
Score (sanyantra, Synset #1) =0
Score(vanaspati, Synset#2) = 7.25
Score (sanyantra, Synset #2) = 0
Score(vanaspati, Synset#3) = 0
Score (sanyantra, Synset #3) = 0
Score(vanaspati, Synset#4) = 0
Score (sanyantra, Synset #4) = 0

As the scores shows vanaspati is matching with Synset #2, to whom it should have matched but sanyantra is not matching with any Synsets. To evaluate the algorithm, we proposed an evaluation criterion

3.2 Corpus Based Approach

In the previous approach, we had used the senses of words given in Shabdaanjali. The second approach for linking Shabdaanjali and WordNet uses example sentences related to each sense given in Shabdaanjali to measure the similarity between sense and WordNet Synset. Using the WordNet, the words in the hypernymy and hyponymy synsets are grouped together to form semantic cluster for each sense of the noun and verb. The verbs co-occurring in the example sentence of a sense is used to obtain the most likely synset for that sense of the noun word. Same thing was done to get most likely synsets for the senses of verb words by using co-occurring nouns. The algorithm uses the corpus based statistical technique for selecting semantic cluster relevant to each sense of the noun and verb. We used British National Corpus (BNC).

3.2.1 Method and Implementation

The intuition of verb-noun relation is that the verb, which is correlated to a noun, also correlates with the hyponymy and hypernymy of that noun. In a given sentence verb can be used to find out the sense of the noun present in the sentence. Same thing is followed for noun-verb relation.

For example, consider the two senses of 'plant', as '*buildings for carrying on industrial labor*' or 'a *living organism lacking the power of locomotion*'. Only the sense of 'buildings for carrying on industrial labor' is available in the context of '*they built a large plant to manufacture automobiles*' because the verb 'built' helps in identifying the meaning of its object plant.

The words co-occurring in the sentence provide essential clue towards the intended meaning of the polysemous words. The verb-noun co-occurring pair determines the meaning of the polysemous nouns and the noun-verb co-occurring pair determines the meaning of the polysemous verbs.

As we have discussed earlier, different synsets of a word conveys different senses of that word. It was the basis of our first approach i.e. dictionary based approach and again it is the basis of this approach.
The method proposed here is basically based on two observations

(a) Different senses of the words appear in different semantic clusters.

(b) The relation between the verb and nouns; verbs which co-occur with a noun also co-occur with the words in the hyponymy and hypernymy synsets of that word.

For example, consider the hyponymy synsets of plant. It is likely to find instances of (build, factory) and (build, manufacturing plant) in British National Corpus and less probable to get instances of (build, acrogen) and (build, aquatic) in British National Corpus. Thus, by combining the lexical knowledge acquired from the IS-A taxonomy and word-word association, a usable estimate of conditional probability can be obtained.

This system takes example sentence of a word from Shabdaanjali as input. Preprocessing was done to find out co-occurring verbs and nouns in the example sentence using Brill Part of Speech Tagger and English Morphological Analyzer. The set of co-occurring verb-noun and noun-verb pairs were obtained from the syntactically tagged British national corpus (BNC). Again preprocessing was done using English Morphological Analyzer. After obtaining the co-occurring verb-noun and noun-verb pairs, probability of each sense with each semantic cluster was calculated.

Conditional probability was calculated as

$$P(\text{noun}|\text{verb}) = \frac{\text{freq (noun, verb)}}{\text{freq (verb)}}$$

The probability of a semantic cluster with the co-occurring verb can be determined as:

$$P(\text{SemCluster}_i \mid \text{verb}) = \frac{1}{Z} \sum_{\text{for all w in Semcluster}_i} P(w \mid \text{verb})$$

Where 'Z' is the normalizing factor taken over all words in all semantic clusters.

$$Z = \sum_{\text{w in all SemCluster}} P(w \mid \text{verb})$$

We have calculated probability for nouns. Probability for verb can be calculated by same method. So, whole algorithm is summarized as:

Preprocess the British National Corpus by using English Morphological Analyzer to obtain verb-noun co-occurring pairs and also noun-verb co-occurring pairs.

For a given word 'w', do the following

- Obtain example sentence related to each sense of 'w' from Shabdaanjali
- Run Brill Part of Speech Tagger and English Morphological Analyzer to get co-occurring nouns and verbs.
- Construct Semantic clusters $SemCluster_i$ for each sense i of 'w'
- For each Semantic cluster obtain the probability with each co-occurring verbs and nouns.
- If this probability is less than defined threshold value, ignore it.
- Select the sense of 'w' for which probability is greater than threshold value

3.3 Combined Approach

This approach is a combination of previous two approaches, Dictionary-based approach and Corpus-based approach. In the Dictionary-based approach, we had used the senses of the Shabdaanjali and in the Corpus-based approach; we had used the example sentences related to each sense. In Combined approach, we have used both the senses and the example sentences.

The combined approach is actually the combination of the results of both algorithms. As we had seen, the results of Dictionary-based approach were much better than the results of Corpus-based approach. So, the results of Dictionary-based approach were used as the base on which results of Corpus-based approach were added.

The algorithm of combined approach is:

For a given word 'w', do the following
  Store Shabdaanjali senses of 'w' in a list SHW
  For each entry 'i' of SHW, do the following
    For each Synset 'j' of 'w', do the following
      Set $Score_{i,j}$ to 0
      Run Dictionary-based algorithm
      If $Score_{i,j}$ is less than 7 then do the following
        Run Corpus-Based algorithm and store result of that algorithm in $CORP\_SCORE_{i,j}$
        If $CORP\_SCORE_{i,j}$ is less than threshold value then set $Score_{i,j}$ and $CORP\_SCORE_{i,j}$ to 0 else set $Score_{i,j} = 7 + CORP\_SCORE_{i,j}$
  return $Score_{i,j}$

4. Experiments and Results

The algorithms have been tested separately for nouns and verbs. Also, separate threshold value was defined for nouns and verbs. The algorithms were tested on 65 randomly chosen nouns and 45 randomly chosen verbs.

In Dictionary Based Approach average precession and recall for noun are 85% and 73% respectively and verb it is 80% and 71% respectively, which shows the effectiveness of this

approach in aligning Shabdaanjali with WordNet. Also, this approach is of very less complexity compared to the other approaches.

The evaluation criterion for Corpus Based Approach is same as we discussed in Dictionary-based approach. This algorithm was also separately tested for nouns and verbs. It was tested on same Data Set on which Dictionary-based algorithm was tested. Also separate threshold values were defined for nouns and verbs. These threshold values had no relation with the threshold values defined in Dictionary-based Approach.

The average precision obtained was 31% and average recall obtained was 95%. After introducing a threshold value of 0.1, the average precision and recall became 49% and 57%.

The average precision obtained was 26% and average recall obtained was 96%. After introducing a threshold value of 0.3, average precision became 39% and average recall became 29%.

These results were not so good but it is said, "Research calculates success in failure". How much this was true in our task, was proved later when we combined both approach i.e. Dictionary-based approach and Corpus-based approach.

Combined approach, using both the strategies, improved the results to 90.75% precision and 64.26% recall for nouns and 89.23% precision and 63.3% recall for verbs.

As seen from the results this combined approach is optimal and suggested for application in word sense disambiguation and machine translation.

The evaluation results are given in following table.

Approach	Lexical Category	Precession (%)	Recall (%)
Dictionary Based	Noun	85.71	72.89
	Verb	80.35	71.42
Corpus Based	Noun	49	57
	Verb	30	29
Combined	Noun	90.75	72.89
	Verb	89.23	63.3

Table1: Results of evaluation

5. Further Development

The above algorithms are also implemented on Roget's thesaurus instead of WordNet. Roget's thesaurus also gives different synset of a particular word with a particular category. Clusters are created from the synsets. Above algorithm are executed over the cluster defined from Roget's thesaurus. As the algorithm create a matrix of weight assigned to each synset for each sense in Shabdaanjali. Now to assign a synset for a sense of Shabdaanjali, it goes through the assignment problem algorithm and assign and assign the best synset for a sense in Shabdaanjali. If all the weights are zero for a sense of Shabdaanjali then no synset is assigned to the sense.

The challenge was to map the correct word from the synset list to the closest synonym of the ambiguous word in the target language. Algorithm gives a sense and the best synset, which fits to the sense.

6. Conclusion

In this paper, we presented a working algorithm that uses both bilingual and monolingual recourses and attempts to map the various senses of a word across languages. Using standard techniques of efficiency testing, we calculated the precession and recall of the algorithm at work and found that the average efficiency stood at 77.15% for noun and 76.27% for verbs.

References

[1] Pianta, E., Bentivogli, L., and Girardi C. MultiWordnet: Developing an Aligned Multilingual database. *1st International Global WordNet Conference,* Mysore, India, 2002.

[2] Carpuat M., Ngai G., Fung P., Church W. K. Creating A Bilingual Ontology: A Corpus-Based Approach for Aligning WordNet and HowNet*, 1st International Global WordNet Conference*, Mysore, India, 2002.

[3] Lafourcade M. and Prince V. Relarive Synonymy and Conceptual Vector, *6th NLP Pacific Rim Symposium,* 2001, Japan. 2001

[4] Narayan D. K. and Bhattacharyya P. Using verb-Noun Association for Word Sense Disambiguation , *Int Conf Knowledge Based Computer Systems*, Mumbai, India, 2002.

[5] Dekang Lin - Automatic Retrieval and Clustering of Similar Words*, COLING-ACL98*, Montreal, Canada, 1998.

[6] *Shabdanjali : A Free English-Hindi Bilingual Dictionary, LTRC, IIIT Hyderabad (2001), http://www.iiit.net/ltrc/Dictionaries/Shabdanjali /dict-README.html*

Corpus-based Study of Lexical Polysemy in Bangla for Application in Language Technology

Niladri Sekhar Dash
Computer Vision and Pattern Recognition Unit
Indian Statistical Institute, Kolkata
Email: niladri@isical.ac.in

1. Introduction

Lexical semantics attests each word with a meaning (either explicit or latent) that plays a pivotal role to differentiate it from its peering members. This meaning is often derived from its etymology. However, context plays a crucial role in partial or total modification of word meaning, as well as in the projection of a new meaning embedded within it.

It is acknowledged that works of language technology (LT) can benefit from the analysis of lexical sense. However, due to complexity of sense analysis and interpretation, the majority of LT systems do not pertain to treatment of lexical sense. Yet, the study of lexical meaning makes significant progress in theories, description, and processing.

Understanding of actual sense of word is very tricky. Scholars have proposed two major approaches to deal with the problem of sense disambiguation of words. The *knowledge-based approach* uses explicit sets of lexicon [**5**], while the *corpus-based approach* uses information obtained from corpus. As we prefer to work with the corpus-based approach, we try to extract information from the analysis of corpus. Information obtained thus will be processed to understand actual contextual sense. To substantiate the argument, supporting examples are drawn from the Bangla written corpus [**6**].

2. Proposition about word meaning

From the very early days of language study, scholars have noted a complex network of relations existing between the word and its meaning. It has been observed that a single idea or sense can be expressed by multiple words; and conversely, multiple related ideas and senses can be expressed by a single word. The former is known as *synonymy*, while the latter is called *polysemy*.

2.1 The modern theory

The view of traditional lexical semantics receives strong criticism from some modern linguists who like to ignore the etymological meaning of words [**4**,**8**,**12**]. They argue that words have no fixed meaning. The meaning arises from the context of use.

Firth [**8**], Gaustad [**10**], Lyons [**12**], Cruse [**4**], and others also put forward similar arguments. According to them, words are meaningful when they are used in context. Their sense variation is extendable, changeable and derivable from their context of use only. What is notable here is that meaning of words is not only associated with etymology and morphological form, but also with syntactic, topical, prosodic, idiomatic, and such similar characteristically observable contexts. This deliberation directs us towards the *non-existential theory of semanticity*, which advocates that contextual information is essential for proper understanding of word sense.

2.2 The Prototype Theory

When we try to decipher the actual sense of a word, we find that the proposition of *non-existence of meaning of words* does not hold good. We cannot raise the issue of sense variation if words do not possess any meaning at all. This underlying fallacy redirects us towards the *Prototype Theory* [**9**] of word meaning. The basic concept of this theory is – a word has a core meaning, but in the context of specific *lexico-grammatical* and *morpho-syntactic* functions of a language, the meaning takes different shades and can be considered to be an independent entity. The core meaning may not be explicit always. It may be expressed through the use of numerous extralinguistic factors. Cognitive linguistics [**3**,**5**] also acknowledges the existence of core meaning which is interlinked to the seeds of multiple senses. If supported by suitable context, this core meaning

may denote many novel senses related to the core sense [**1**].

This process works in every living language because language users, rather than coining a whole new set of words, often tend to use a particular word to denote similar ideas, concepts, items or objects. Although the reason for this is unknown, it is probably due to the lack of a suitable vocabulary as well as the presence of a conceptual interface between the old and the new ideas motivate language users to use single words to refer to similar senses.

3. Issue of sense variation

Once we agree with the argument that in *polysemy* several related senses are packed together under the same head [**7**], we realise that the question of sense variation becomes an important issue. It means that a word can have multiple senses, which are related by *semantic extensions*. Thus, the Bangla word *kathā*, due to sense variation can denote *word, statement, description, narration, story, opinion, promise, excuse, context, conversation, suggestion, provocation, prescription, order, request, compulsion, explanation,* and other similar verbal expressions.

Once we scrutinise the examples cited above, the phenomenon of sense variation reveals a network of intricate relations among the senses, which hardly comes to the surface. We also identify the following features of sense variation that provide clues for designing systems for word sense disambiguation.

- Every word has a (explicit or implicit) core sense.
- Sense changes due to context of word use, and
- Every sense of a word has a relation with the core sense.

Corpora help us locate contexts methodically, and classify and access them for necessary information. Availability of electronic corpora makes the task of word sense disambiguation much easier, which was not possible before with intuitive evidence. Corpus-based empirical linguistics excels over the intuition-based generative linguistics not only in supplying a wider spectrum of contexts of word use, but also in providing necessary contextual information for understanding the variation of senses a word can denote.

Before we proceed further, it is important to distinguish between *polysemy* and *homonymy* as discussed in the next section.

4. Polysemy and homonymy

Polysemy is often associated with *homonymy*, because the distinction between the two is not always clear. In polysemy, words show sense variations due to context, while in homonymy different unrelated meanings or senses share the same orthographic forms [**7**]. Homonymy is expressed either in 'similar spelling different meaning scheme' (*māl*, *jin*, etc.), or 'dissimilar form similar utterance scheme' (*dīn* 'poor' and *din* 'day', *śab* 'dead body' and *sab* 'all', etc.). To illustrate this, let us look into the use of *māl* in the following sentences obtained from a corpus.

(1) *Mālbhūmir khub kāche śaharṭi abasthita.*
"The town is located very near the plateau."

(2) *Sāper oṣudh āche māler kāche.*
"The anti-venom is with the snake-charmer."

(3) *Tār kustir hātekhaṛi hayeche māleder ākhṛāy.*
"He learned wrestling at the wrestlers' camp."

(4) *Tār galāy muktār māl, ār komare bāgher chāl.*
"A pearl necklace is on his neck, and a tiger-skin is on his waist."

(5) *Roj sandhāy tār du peg māl cāi.*
"He needs two pegs of liquor every evening."

(6) *Se etaguli māl ekā baite pārbe nā.*
"He can't carry so many goods alone."

Apparently it seems that *māl* is a polysemous word, which is used in six different senses in six different contexts. Actually, they are six different homonyms, which display orthographic similarity. They differ both in meaning and etymology (Table 1) in the following way:

No	Word	Meaning	Source	Etymology
1	māl	highland	< Skt.	mā + -la
2	māl	snake charmer	< Skt.	mal + -a
3	māl	wrestler	< Skt.	malla
4	māl	garland	< Skt.	mālā < mālya
5	māl	liquor	< Pers.	māl
6	māl	item	< Arb.	māl

Table 1: Homonym with different origins

Some general criteria are provided in Table 2 to explore the differences existing between the two.

Criteria	Polysemy	Homonymy
Existence	Word level	Word level
Structure	Single form	Similar forms
Spelling	Do not vary	May vary
Pronunciation	No variation	Negligible variation
Sense variation	Due to context	Due to etymology
Context	Plays vital role	Not relevant

Table 2: Polysemy vs. homonymy

The distinction between the two is not straightforward, since words that are etymologically related can, over time, drift so far apart that the original semantic relation is no longer recognizable . A homonym resulting from an accidental convergence of forms is reinterpreted as a case of polysemy [**11**]. Therefore, we need to separate the principled system of multi-semanticity (polysemy) from the accidental convergence of orthography (homonymy) for various applications in linguistic description and language technology.

5. Lexical polysemy in Bangla

Almost all languages have a set of words, which are capable of conveying multiple senses. Thus, lexical polysemy becomes an essential property of a language, where numerous ideas and senses are represented by a bounded (though large) number of words available in it. In the Bangla corpus of around 5 million words, nearly 10% of the vocabulary exhibit two or more sense variations. The verbs *khāoyā* has 40+, *kāṭā* has 20+, *mārā* has 25+; adjectives *khārāp* has 30+, *baṛa* has 20+, *choṭa* has 15+; nouns *māthā* has 30+, *bhāb* has 25+, *mukh* has 20+ senses. While describing the nature of lexical polysemy in Bangla, we observe the following

(a) The number of polysemous words in the Bangla corpus is quite large (nearly 10% of the words).
(b) These words are mostly nouns, adjectives or verbs.
(c) The most commonly used words are mostly polysemous in nature.
(d) A word can remain polysemous in spite of the change of its lexical category.

6. Factors behind lexical polysemy

There exist some linguistic and extralinguistic factors which may decide which words are polysemous:

- The occurrence of the word in different lexical collocations helps generate new senses. Here, we observe a kind of semantic shift when a particular word (W_1) collocates with another word (W_2) to generate an altogether new sense. The core sense of the word in question (W_1) is changed due to its collocational relation with another word (W_2). It is difficult to understand such variation of the sense of W_1 if we do not analyse and associate the meaning of W_2 with that of the W_1.
- Change of lexical category is another important factor for polysemy. It causes words to generate new sense, which is not different entirely from the core sense, is somewhat different. Such conceptual expansion adds an extra shade to the actual core sense. For example, *chāṛā* is usually used to mean 'without', which is lexically an adverb (ADV) derived from the verb root (FV-Rt.) √*chāṛ* 'to free'. The word is also used as noun (NN) to mean 'a matured female calf', and as adjective (ADJ) to mean 'something which is free'. In each lexical category, the word carries its latent sense (sense of separation) which is originally found in the verb root. Fig. 2 can show how change of lexical category can cause variation in sense of a word.
- Extralinguistic factors (society, culture, geography, ethnography, demography, history, etc), which have indirect connection with the words under consideration, may cause polysemy.

How extralinguistic factors cause polysemy is difficult to analyze, but the explanation given by Backhouse [**2**] is:

> "… language is used in the world, and lexical items relate to aspects of this world: in particular, lexical items are applied to extralingual categories of entities, qualities, actions, events, and states, and the relation between an item and such categories … is normally understood as constituting a central part of its meaning."

7. Types of lexical polysemy in Bangla

Polysemous words may be uninflected, or may be tagged with inflection markers and affixes.

7.1 Polysemy in non-inflected forms

The most common type of polysemy is observed among non-inflected nouns, verbs, and adjectives. The Bangla corpus contains a large number of such

words with polysemous make-up. In Table 3, we present the 10 most polysemous adjectives.

Adjective	Core sense	Senses
kā®cā	raw	30+
pākā	ripe	25+
khārāp	bad	30+
garam	hot	25+
baṛa	big	20+
manda	bad	20+
miṣṭi	sweet	20+
mukta	free	20+
laghu	light	20+
śakta	hard	20+

Table 3: Adjectives sense variations

All the words listed above are simple in form while empirical analysis of corpus shows that these are highly recurrent in use. This validates our stand that the most commonly used words are mostly polysemous in nature. As it is not possible to present all the words with their total range of sense variations with reference to their actual contextual occurrence, we select only the two most polysemous words (a noun and an adjective), and present their possible list of sense variations obtained from the Bangla corpus.

Word : māthā
Lexical class : Noun
Core meaning : 'head'
No. of sense variations : 30+

***Examples**:* mānuṣer māthā *(human head),* gācher māthā *(tree top),* chātār māthā *(useless thing),* ṭebiler māthā *(top of a table),* grāmer māthā *(head of a village),* pāhāṛer māthā *(mountain peak),* āṅguler māthā *(finger tip),* jaler māthā *(surface of water),* nadīr māthā *(source of river),* rāstār māthā *(end of road),* cār māthār moṛ *(crossing of four roads),* kaci māthā *(tender mind),* māthā muṇḍu *(head and tail), etc.*

7. Conclusion

We support Moravesik to argue that it is not necessary to define all possible and potential variations of sense of words. Lexical polysemy is a vital element in a natural language, and if for each polysemous reading we introduce a new semantic entry in the lexicon, we will burden the lexicon as well as the language learners excessively, and lose productivity and flexibility of our language use. The flexibility is needed because at any given point of time, a language may not mark out each meaning of a polysemous word sharply. Lexical polysemy can leave semantic interpretation of a word in a state where various shades may be added in context.

However, in lexical semantics, we are concerned with tackling the problem of lexical polysemy for the following reasons. First, we need to understand the phenomenon in order to perform word sense disambiguation (of use in machine translation, information retrieval, content analysis, tagging, natural language understanding, aligning bilingual corpora, etc). Secondly, we need systematic information about polysemous words for using it for lexical resource development (wordnets, dictionaries, lexicons), and language teaching.

9. References

[1] Alanko, P.K. "Mechanisms of semantic change in nouns of cognition: a general model", in Coleman, J. and Key, C.J. (eds.) *Lexicology, Semantics, and Lexicography.* Amsterdam: John Benjamins. 35-52. 2000.

[2] Backhouse, A.E. *The Lexical Field of Taste: a semantic study of Japanese taste terms.* Cambridge: Cambridge University Press. 1994.

[3] Coleman, J. and Key, C.J. (eds.) *Lexicology, Semantics, and Lexicography.* Amsterdam: John Benjamins. 2000.

[4] Cruse, A. *Meaning in Language: An Introduction to Semantics and Pragmatics.* Oxford: Oxford University Press. 2000.

[5] Cuyckens, H. and Zawada, B. (eds.) *Polysemy in Cognitive Linguistics.* Amsterdam: John Benjamins. 2001.

[6] Dash, N.S. and Chaudhuri, B.B. "The process of designing a multidisciplinary monolingual sample corpus". *International Journal of Corpus Linguistics*. 5(2): 179-197. 2000.

[7] Fellbaum, C. "Autotroponymy", in Ravin, Y. and Leacock, C. (eds.) *Ploysemy: Theoretical and Computational Approaches*. New York: Oxford University Press. 52-67. 2000.

[8] Firth, J.R. "Modes of meaning", in *Papers in Linguistics*. Oxford: Oxford University Press. 1957.

[9] Geeraerts, D. *Diachronic Prototype Semantics: A Contribution to Historical Lexicology.* Oxford: Clarendon Press. 1997.

[10] Goustad, T. "Statistical corpus-based word sense disambiguation: pseudo-words vs. real ambiguous words", in *Companion Volume to the Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*. 24-29. 2001.

[11] Leech, G. *Semantics.* Middlesex, Middlesex, England: Penguin Books Ltd. 1974.

[12] Lyons, J. *Structural Semantics.* Cambridge: Cambridge University Press. 1963.

MACHINE TRANSLATION

Annotated Generation of the Hindi Case System in an Interlingua based MT Framework

Debasri Chakrabarti, Sunil Kumar Dubey, Pushpak Bhattacharyya.
Computer Science and Engineering Department,
Indian Institute of Technology, Bombay,
Mumbai, 400076, India.
debasri,dubey,pb@cse.iitb.ac.in

1. Introduction

The case markers play an important role in the structure of a sentence to impart the meaning and naturalness. Consider the following example

1. *मोटे तौर पर कृषि भूमि की जुताई, फसलों की रुपाई, कटाई, पालतू पशु प्रजनन, पालन, दुग्ध-व्यवसाय और वनीकरण सम्मिलित होता है ।

In a broad sense, agriculture includes cultivation of the soil and growing and harvesting crops and breeding and raising livestock and dairying and forestry.

The meaning of the above sentence becomes indiscernible if कृषि is not followed by the case marker में.

The present paper describes the structure of the case system in Hindi, the selection of cases according to morphology and semantics and the application of this knowledge in the UNL based English-Hindi MT.

2. Theoretical Approach: The case system in Hindi

Hindi is a language of the New Indo-Aryan family spoken in the parts of Northern India. It is an inflectional language. One of the most notable features of the nominal system of Hindi is its rich sub-system of case. The relation between the words of the language is denoted through the case markers. Consider, the following example,

2 राम ने रवि को किताब दी।
Rama Erg Ravi Dat book Nom give + past.

Each of the three nouns in the above sentence bears a distinct case. Erg stands for ergative, Dat for dative and Nom denotes nominative. Hindi has the following cases: nominative, accusative, instrumental, dative, ablative, genitive and locative. Each case is represented in the language by a specific case marker. In Table 2.1. the language universal case feature is shown.

Table 2.1. Language Universal Case Feature

Case	Conditions
Nominative (NOM)	case of the subjects. In a language if there are two distinct cases for the subjects, one inflected and the other without inflection then NOM refers to the uninflected one.
Ergative (ERG)	the inflected case associated with the subject
Accusative (ACC)	case attached with the object
Dative (DAT)	case of goals/ recipients.
Instrumental (INS)	case of instruments used to accomplish an action
Genitive (GEN)	case of possessors
Locative (LOC)	case of physical place

Apart from the above generalization, every language has its idiosyncratic features [2]. Hindi, like other South Asian languages, has no one to one mapping between the grammatical function and case marking. A subject may be nominative and / or ergative, dative, instrumental *etc* [3]. The object may be either accusative or nominative. Thus, case marking in Hindi is overloaded. A particular grammatical function can be mapped to different case markers. The case structure of Hindi is shown in Table 2.2. It accounts for the case markings along with their semantic and syntactic properties.

It is transparent from Table 2.2. that case marking is a complex phenomenon in Hindi. Apparently this seems to be quite erratic in nature. Detail study reveals that there is a valid logic beneath this behavior and this depends to a large extent upon the morphological and semantic features of the language. For the current purpose the work is only confined to the nominative, ergative and accusative cases.

2.1. Nominative ~ Ergative alternation in the agent position

In Hindi the agent of an action may bear either nominative case or ergative case. It has been found that ergative case appears in Hindi with the verb in its simple past form and in perfective aspect this is

Table 2.2. Case features of Hindi

Case	Markers	Conditions	Example
Nominative	Ø	a. Subject	*राम* आम खा रहा था।
		b. Inanimate primary object	राम *आम* खा रहा था।
Ergative	ने	a. Agentive subject with perfective aspect	*राम ने* श्याम को किताब दी थी।
		b. simple past	*राम ने* किताब पढ़ी।
Accusative	को	a. Animate primary object	राम ने *सीता को* देखा।
		b. Definite, Inanimate primary object	राम ने *उस किताब को* पढ़ा।
Dative	को	Goal of the sentence	राम ने *सीता को* किताब दी।
Instrumental	से	a. Instrument	राम ने *चाकू से* फल काटा।
		b. Intermediary agent[cause]	राम ने *सीता से* चिट्ठी लिखवाई
		c. Denoted agent of Passive	*राम से* खाना नहीं खाया गया।
Ablative	से	Source	*पेड़ से* पत्ते गिर रहे हैं।
Genitive	का, की, के	a. Possessor [involving ownership of something]	राम *की किताब* अच्छी है।
		b. Relationship to somebody	*राम का* भाई अच्छा है।
Locative	में	a. In, Within	राम *दिल्ली में* रहता है।
	पर	b. On, at	राम *पेड़ पर* चढ़ गया।

irrespective of any tense. Consider the following examples,

3 राम ने रवि को पीटा।
Rama erg Ravi acc beat+past.

4 राम ने रवि को पीटा था।
Rama erg Ravi acc beat+past perfect.

5 राम ने रवि को पीटा है।
Rama erg Ravi acc beat+present perfect

6 राम ने रवि को पीटा होगा।
Rama erg Ravi acc beat+future perfect

Based on this observation it has been concluded that there is a correlation between the ergative case and the aspectual property of the main verb. Moreover, morphologically overt on the verb. The form of the main verb bearing the above mentioned tense and aspect will be morphologically shown as:

V + आ → ने (cf. 3)

V + आ + (Tense morphology) → ने (cf. 4, 5 & 6)

There are a number of cases attested in the language with the above mentioned aspectual feature in the main verb but with the nominative case on the agent. This enabled the scholars to claim that the NOM~ERG alternation in the agent is largely motivated by the transitive-intransitive distinction and associated ergative with the transitive verbs. This is a language universal feature. Following this view some scholars in Hindi too argued for the same [4] but this has turned false for the language. Hindi shows three types of patterns independent of transitivity: only those that, given the required aspectual conditions, take a) only NOM agents, b) only ERG agents c) either NOM or ERG agents. Consider the following examples which stand for the above argument.

7. i) राम गिरा।
Rama +nom fall + past.
ii) *राम ने गिरा।
Rama erg fall + past.

8. i) राम ने प्रतीक्षा की।
Rama erg wait + past.
ii) *राम प्रतीक्षा किया।
Rama +nom wait + past.

The verbs in the example from 7-8 are intransitive. The same is true with the transitive verbs also. Ergative case in Hindi is semantically driven. If an action is performed deliberately by an agent then the ergative case occurs whereas if an action takes place non-deliberately then nominative case is attached to the agent. Logically no one will *fall down* deliberately but the action of *breaking* is related to agent's conscious choice. This view is extended for some non-sentient volitional agents too. There are some inanimate agents that can perform action deliberately. For instance,

9. हवा ने पत्ते बिखेर दिये।
wind erg leaves scatter + past aux.
'The wind had scattered the leaves'.

Therefore, it can be rightly summarized that in Hindi the NOM~ERG alternation is subject to the semantic property of the verb. This property is termed as the *conscious choice*. Given a verb in its perfective aspect the agent will choose the erg case if the action is performed deliberately.

2.2 Accusative ~ Nominative alternation in the object

Primary objects in Hindi are either acc with the marker
को (ko) or are nom (uninflected). For example,

10. राम ने चावल खाया।
    Rama erg rice + nom eat+ past.
11. राम ने रावण को मारा।
    Rama erg Ravana acc kill + past.

A widely accepted generalization with regard to objects in Hindi is that the canonical case for animate objects is acc and for inanimate object is nom. This is true for all the South Asian languages. There are counter examples of this generalization. Cases are attested in the language where acc case appears with the inanimate objects also.

The above stated phenomenon, though complicated is regular in Hindi. Various methodologies are taken to handle this in the UNL based generation system.

In the next section a brief overview of the UNL system and the English- Hindi MT system is given.

3. Universal Networking Language UNL

The English-Hindi MT system described here is based on an Interlingua, popularly known as Universal Networking Language (UNL).

UNL is an electronic language for computers to express and exchange information [5]. UNL system consists of *Universal words (UW)*, *relations, attributes*, and the *UNL knowledge base (KB)*. The UWs constitute the vocabulary of the UNL, relations and attributes constitute the syntax and the UNL KB constitutes the semantics. UNL represents information sentence-by-sentence as a hyper-graph with concepts as nodes and relations as arcs. UWs can be annotated with attributes like *number*, *tense etc.*, which provide further information about how the concept is being used in the specific sentence. Any of the three restriction labels- *icl, iof* and *equ-* can be attached to an UW for restricting its sense.

3.1 UNL based generation system

Under every MT system there are two components: analysis and generation. For the present purpose only the generation system is explained here. It generates target sentences of a native language from UNL expressions by applying generation rules [6]. In addition to the fundamental function of the generation engine, it checks the formats of the rules and the UNL expressions as input, and outputs messages for any errors. It also outputs the information required for each stage of generation in different levels. The generation system consists of a UW-dictionary and rule base.

Description of the rule base. There are various types of rules in the rule base. One frequently used rule for the Hindi generation is the *left insertion rule*. Hindi is a SOV language. Hence, a child node is mostly always inserted to the left of the parent node given in the UNL expression.

Format for the Left Insertion rule of a node is shown below:

:"<COND1>:<ACTION1>:<RELATION1>:<ROLE1>"{
<COND2>:<ACTION2>:<RELATION2>:<ROLE2>}

This type of rule inserts the child node linked by the relation <RELATION1> or <RELATION2> to the left of the parent node. This will become clear with a real example.

Interpretation of a rule. An example of left insertion rule is interpreted below:

:"<agt:+blk,+agt,+!ne,+sufc:agt:"{V,>agt,@past,DL
BRT-ACT,^@progress:+!agt::}P242;

In the above rule curly braces are used to indicate the parent node and double quotes for a child node. *V,>agt,@past,DLBRT-ACT,^@progress* states the condition of the parent node. Similarly, *<agt* states the condition of the child node. The relation between the parent and the child node is *agt*. Firing of this rule will insert *!ne* to the child node. This will result finally in attaching ने case marker to the child node.

4. Methodology and Implementation

Various methodologies are used in generating the appropriate case markers for the *agt* and *obj* relations in UNL system. These are discussed below

4.1. Rules to handle case markers on the agent

The ergative marker is dependent on the semanticity of a particular verb. This information is encoded in a verb hierarchy by the attribute [DLBRT-ACT] which stands for *deliberate action*. In the hierarchy built for the verbal concepts of the Hindi both syntactic and semantic information are given [7]. As verbs assign case markers to the nouns thus, it is decided that the case information will be coded in the hierarchy through syntax or semantics. The verb पीटना *to beat* will have the semantic attribute [DLBRT-ACT] where as in गिरना *to fall* it will be absent. The suffix -आ gives the information

of the tense and aspect. In the UNL system simple past tense and the perfective aspect will be represented as *@past* and *@complete* respectively. The generation system will now check @past or *@complete* with DLBRT-ACT to generate the ergative case. This approach will also help to generate the correct output of the sentences like 13. As the verb will be attributed semantically thus, sentient or non-sentient agent will make no difference.

Following are the rules made for the generation system to handle the case of the agent.

R1→
:"<agt:+blk,+agt,+!ne,+sufc:agt:"{V,>agt,@past,DLBRT-ACT,^@progress:+!agt::}P242;
R2→
:"<agt:+blk,+agt,+sufc:agt:"{V,>agt,@past,^@progress:+!agt::}P241;
R3→
:"<agt:+blk,+agt,+!ne,+sufc:agt:"{V,>agt,@complete,DLBRT-ACT,^@progress:+!agt::}P242;
R4→
:"<agt:+blk,+agt,+sufc:agt:"{V,>agt,@complete,^@progress:+!agt::}P241;

The above rules are of insertion type. Application of these rules will help to generate the correct output in the UNL system for the examples from 3-13. Example 3 will be handled by R1. R2 will generate sentence 7 correctly. Examples from 4-6 will be handled by R3 and R4 is for a sentence like 18. राम गिरा होगा।

Ram+ nom fall+ future perfect
Ram must have fallen down.

4.2. Rules to handle case markers on the object

The accusative marker on the primary object is pertinent to animacy. The three cases defined in section2.2 are handled in the generation system by the following rules.

R5 →
:"<obj,INANI,MALE,^V,^SCOPE:+obj,+sufc,+blk:obj:"{>obj:+!obj,+male::}P160;
R6 →
:"<obj,INANI,FEMALE,^V,^SCOPE:+obj,+sufc,+blk:obj:"{>obj:+!obj,+female::}P160;
R7 →
:"<obj,ANIMT,MALE,^V,^SCOPE:+obj,+sufc,+blk,+!ko:obj:"{>obj:+!obj,+male::}P160;
R8 →
:"<obj,@def,MALE,^V,^SCOPE:+obj,+sufc,+blk,+!ko:obj:"{>obj:+!obj,+male::}P163;

R5 and R6 together generate the correct output for the inanimate objects (cf. 14). The animate objects are handled by R7. R8 will give the correct output for the inanimate definite object. The accusative marker on the inanimate object is taken care of by stipulating the attribute *@def* on the object. *@def* stands for *definite* and this attribute is obtained from the UNL expression which is generated by the English analyzing system. The generation system will check the @def feature to attach को with the inanimate object.

The number at the end of a rule is the priority marker. These show the order in which a particular rule is fired. More attributes on nodes will dictate higher priorities of a rule.

5. Result, Conclusion and Future Work

It is apparent from the above study that handling of the case markers in natural language generation is a significant and challenging task. One cannot build a robust system without studying the case structure minutely. The discussed methodologies provide encouraging result and lend naturalness in the generation of the Hindi sentences. The work will be further enhanced by studying all the case markers with respect to the UNL relations.

References

[1]. Tara Warrier Mohanan, *Arguments in Hindi,* A Dissertation, Dept. of Linguistics, Stanford University, 1990.

[2]. Shachi Dave, Jignashu Parikh, Pushpak Bhattacharyya, "Interlingua Based English Hindi Machine Translation and Language Divergence", Journal of Machine Translation, vol 17, 2002.

[3]. Pritha Chandra, *Dative Structure in Hindi: A Minimalist Study,* Jawaharlal Nehru University, New Delhi-110067, India, 2000.

[4]. Yamuna Kachru, *An Introduction to Hindi Syntax,* A Research Paper, Dept of Linguistics, University of Illinois, Urbana, USA, 1966.

[5]. Hiroshi Uchida, Meiying Zhu, Tarcisio Della Senta, *Universal Networking Language, A Gift for a Millennium*

ANUBAAD – A Hybrid Machine Translation System from English to Bangla

Sudip Naskar, Diganta Saha, Sivaji Bandyopadhyay
Computer Science & Engineering Department
Jadavpur University, Kolkata – 700 032, INDIA
sudip_naskar@hotmail.com, neruda0101@yahoo.com, sivaji_ju@vsnl.com

1. Introduction

All manuscripts must be in English. A sentence can have three types of phrases - Noun Phrase, Verb Phrase and Adverbial Phrase. A sentence may also include phrase prepositions and phrasal adjectives (as a part of noun phrase or prepositional phrase). The Noun and the Adverb phrases are translated using Syntactic Example bases. Verb phrase translation scheme is Rule Based (RB) and uses the Morphological Paradigm Suffix Table. In order to translate properly we also need to identify and translate gerunds, infinitives and participles.

The system uses Proper Noun, Acronym, Abbreviation, Phrase prepositions and Idiom dictionaries besides the standard lexicons and several Example bases. We have also used tables that contain the proper nouns, acronyms, abbreviations, phrase prepositions and idioms in English and the corresponding translation in Bangla. For translating idioms we make use of an example base that contains both literal and pattern examples.

2. Morphological Analysis

The input sentence is analyzed to identify the various words/terms and their category. Output is a set of words/terms associated with POS, their category, suffixes attached to them and other grammatical and semantic information. Sentence segmentation, expansion of contracted words (as in *I'd*) and tokenization are performed in this phase. Multiword expressions or terms are identified and treated as a single token. Sequences of digits and certain types of numerical expressions, such as dates and times, money expressions, and percents are also treated as a single token.

3. Shallow Parsing

We have defined a formal grammar for each type of phrases that identify the phrase structure based on the POS information of the lexical tokens. A set of position-based and structure-based rules are applied to words that have multiple POSs in order to reduce the possibilities. Finally, the correct POS might have to be manually provided for such a word.

The phrases, that are identified, are tagged during shallow parsing. These tags include all the necessary information that might be needed to translate these phrases and perhaps resolve ambiguities in other phrases.

4. Example Base and Phrasal Template Matching

We make use of an Example Base to translate noun phrase and adverbial phrase. This contains translation examples (phrasal templates) that store the POS of the constituent words along with necessary semantic information. These translation rules are effectively transfer rules that are stored in the example base instead of explicit coding. This adds flexibility – new rules can be added and existing rules can be modified easily. Here are some examples of noun phrases:

[1] < art $ a / an >< noun & *singular, human, nominative* > ←→ < একজন > < noun >

[2] < art $ the >< noun & *singular, human, objective* > ←→ < noun > <- টিকে >

[3] < art $ a / an >< adj >< noun & singular, *inanimate, objective*> ←→ < একটি > < adj > <noun>

An example of an adverbial phrase is

[4] < prep $ to / at / in >< art $ the >< noun & *singular, place* > ←→ < noun > < - ে / তে >

Let us consider an example sentence: "A *man had given the boy a good book in the school.*". From the above sentence the shallow parser will generate following three tagged noun phrases:

[1] < art & a >< noun & man & *common, singular, third person, masculine, animate, human, nominative, person*>,

[2] < art & the >< noun & boy & *common, singular, third person, masculine, animate, human, objective, person*> and,

[3] < art & a >< adj & good >< noun & book & *common, singular, third person, neuter, inanimate,*

objective, object> corresponding to the three noun phrases *A man, the boys* and *some books* respectively and the following tag is generated for the adverbial phrase *in the school*:

[4] < prep & in >< art & the >< noun & school & *common, singular, third person, neuter, inanimate, nominative, place*>

The template-matching module will find out the matches in the phrasal template example base corresponding to the three NPs and the ADVP respectively. Once appropriate matches have been found for noun and adverbial phrases, translating them is easily done by replacing the POS variables on the target side of the examples with target language meaning of the matched word and then adding the suffixes, if any.

5. Verb Phrase Translation

Translation of verb phrases is treated in a different way. Root verbs in Bangla are classified into a number of groups based on their spelling structure. The different inflections that are to be added to the root verbs are the same for all the verbs belonging to the same verb group with some minor variations. Depending on the person information of the subject of the verb, the respect shown to the subject, the tense of the verb, the spelling structure of the Bangla root verb and the type of the sentence, appropriate verb suffixes are added to the root verbs. These verb suffixes also change from the Classical to Colloquial form of Bangla. Separate Morphological Paradigm Suffix tables have been developed for each of the verb groups. For example, in the sentence "A *man had given the boy a good book in the school.*", the verb phrase is translated as 'দিয়েছিল' ['দি' (Bangla root verb corresponding to English root verb *'give'*) + 'য়েছিল' (suffix corresponding to third person and past perfect tense for this verb group and assuming medium respect to be shown to the subject)] in Colloquial form. If the subject is to be shown high respect, 'য়েছিলেন' would have been added to the Bangla root verb.

We have considered all types of verb phrases. The participles, gerunds and infinitives are also handled properly. Translation of verb phrases from English to Bangla is almost complete.

6. Combination of Phrasal Translations

After the phrases have been translated individually, we use some heuristics to combine the phrase translations. The fact that Bengali is a relatively free word order language makes the problem somewhat easier. No matter how the system combines the phrase translations, the meaning is still the same, but may be somewhat less acceptable.

We have employed the most acceptable heuristic. The subject comes first, followed by the indirect object, if any, followed by the direct object, if any. And the verb phrase is placed at the end of the Bangla sentence. The relative order of the noun phrases usually remains unchanged in the Bangla translation. The adverbial phrases that follow a noun phrase (or verb phrase) are placed before the noun phrase (or verb phrase) in reverse order, in the Bangla translation. From the example sentence - "A *man had given the boy a good book in the school.*" the phrases are translated as:

A *man* ←→ একজন লোক
had given ←→ দিয়েছিল
the boy ←→ ছেলেটিকে
a good book ←→ একটি ভাল বই
in the school ←→ বিদ্যালয়ে

Following the above heuristics, the Bangla translation of the example input sentence will be generated as - "একজন লোক ছেলেটিকে বিদ্যালয়ে একটি ভাল বই দিয়েছিল".

7. Conclusion

The work is the first English to Bangla machine translation system. The present system works with monosemous words and can translate simple and compound sentences (assertive, negative and interrogative) in active voice. We are currently working with clauses (for complex sentences), passive voice sentences, sense disambiguation and term identification.

8. References

[1] S. Bandyopadhyay. An Example Based MT System in News Items Domain from English to Indian Languages, *Machine Translation Review*, (12):7-10. 2001

[2] S. Bandyopadhyay, State and Role of Machine Translation in India, *Machine Translation Review,* (11):25 - 27, 2000

[3] S. Bandyopadhyay, Teaching MT – An Indian perspective, *Proc. 6th EAMT Workshop on Teaching Machine Translation*, UK, 2002, pp. 13-22

[4] S. Bandyopadhyay and D. Saha. Anaphora / Coreference Identification in Machine Translation of News Headlines. *Proc. DAARC 2002, Lisbon, Portugal.*

[5] S. Naskar, S. Bandyopadhyay. Translation of Verb Phrases from English to Bangla, *CODIS-2004*, Kolkata, pp. 582 - 585.

The Root and Epistemic Possibility in Hindi and Bangla

Jayshree Chakraborty
Dept of HSS, IIT Kharagpur 721302
(On leave from K.M. Institute, Dr. B.R. Ambedkar University, Agra 282004)
j_chakraborty2000@yahoo.com

1. Introduction

The present study is an attempt to explain the semantics of some modal auxiliaries in Hindi and Bangla in order to relate the epistemic possibility to root possibility. In section 1 , for the convenience of understanding the entire modal perspective of a language , I first explain the concept of modality and also briefly describe its types. In section 2 , the relevant data from the two languages are examined and analysed and the results of the analysis are discussed keeping in view the objective of the study , i.e., to establish a connection between epistemic and root possibility. In the concluding section, I try to connect the results of the present study to a generalization regarding the nature of the languages with respect to the dominance of tense, aspect and mood.

Section 1

Possibility and necessity are the terms related to the notion of modality in language. In order to understand the semantic concept of possibility and the modal auxiliaries and other syntactic devices expressing possibility, it is necessary to define the domain of modality in language. Modality is related to one's conceiving of things being 'otherwise', i.e., being other than what they actually are. 'otherwise' here means one's imagining or thinking of things being true or false in some possible world other than the actual world , or in the same actual world at some other point in time.

Modality maybe of different types according to the event or proposition's being qualified by a particular set of affairs or a set of principles. The three major dimensions of modality may be recognized as :

(i) epistemic modality
(ii) deontic modality
(iii)dynamic modality

Each one of these types of modality is relativized to a particular set of laws. For example, if one says 'it may rain in the evening' his utterance may be qualified by the rational inference from some evidence like there may be cloud in the sky or he might have come across some weather forecasts etc. Thus in epistemic modality the speaker's interpretation of the world conforms to the rational laws of inference or deduction. In deontic modality the event is relativized to the social or institutional laws. Obligation, permission and order etc. come under deontic modality. The third type of modality which is known as dynamic modality refers to the relationship between a set of circumstances and unactualized events in accordance with natural laws.

In all the three instances of modality mentioned above it may be noted that Hindi and Bangla both employ the same modal auxiliary. In the case of Hindi it is 'sak' whereas in the case of Bangla the auxiliary is 'paar'. Thus it is evident that 'sak' and 'paar' both cause ambiguity and mostly this ambiguity is resolved in the context.

The subject of the present study is , however, not related to this. 'sak' and 'paar' both the auxiliaries also express another kind of modality which is root modality. The subject of the present study is to explore the conceptual notions relevant for expressing different dimensions of root and epistemic modalities. We can see that in Hindi the domain of modality is more finely divided according to the notion of possibility with respect to actualized or unactualized events. In our study we find that these notions are realized mostly through lexical devices in the language. In Bangla, these distinctions are mostly shown with the help of tense markers.

Section 2

In this section we first examine the data for root and epistemic modals and identify those areas which anticipate more observation for proper understanding and detailed description of the semantics of the modal apparatus of Hindi and Bangla. Consider, for example, the following sentence in Hindi:

1. *vah ek minute mein sau miTar dour saktaa hei.*

The sentence gives two interpretations namely:

(a) He can run 100 metres in one minute , and(b) He may run 100 metres in one minute.

In 1(a)the modal auxiliary gives a root modal interpretation modifying the subject of the sentence, indicating his ability whereas in 1(b) , the modal auxiliary expresses the speaker's opinion showing the possibility of the proposition ' he run 100 metres in one minute'. Steele (1975)is of the opinion that the ability interpretation of 'can' cannot be called a modal expression because it does not 'indicate the possibility of the situation which the sentence describes , but rather the potential'.

I wish to argue for the view that 'potential itself is a kind of possibility as possibility can be 'theoretical' and 'factual'(Leech 1971) and I want to support my argument with suitable examples from Hindi-a language which very explicitly makes this distinction in other areas of modality as well. Leech (1971)in his study on the English verb makes a contrast between theoretical and factual meaning in the following sentences:

2. The pound can be devalued.
3. The pound may be devalued.

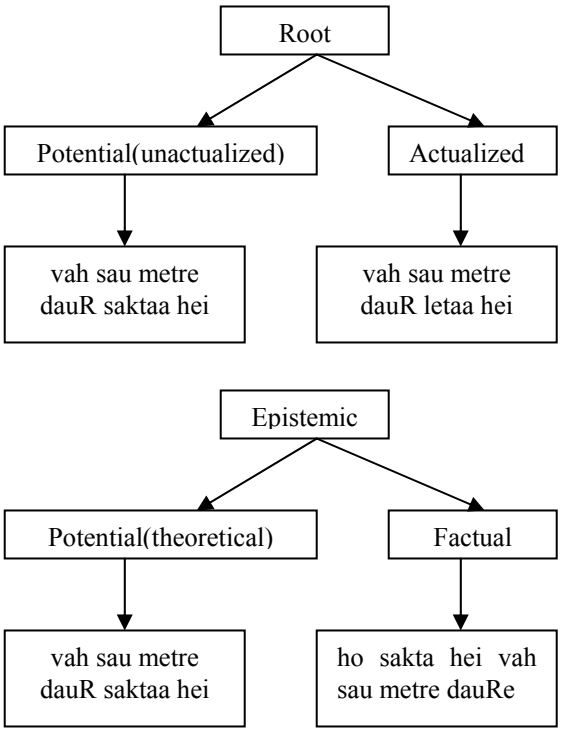
In sentence 2. he says that devaluation is treated as an idea giving the interpretation 'it is possible for the pound to be devalued' whereas sentence 3. treats devaluation as a possible 'fact' , giving a factual interpretation ,i.e. , 'it is possible that pound will be devalued'.

I argue that the 'potential' interpretation assigned to 'root' modality in sentence 1.could also be extended to the theoretical possibility indicated in sentence 2., the Hindi equivalent of which will be ' pound kaa devaluation ho sakta hei'. 'sak'(just like its role in the root interpretation) suggests here also that pound has the ' potentials'(the scope) for being devalued. Sentence 3. on the other hand,gives a factual interpretation and hence has a different equivalent in Hindi where the possibility is indicated through a subjunctive verb. Thus the Hindi equivalent of 3.will be :

3 (a) ho sakta hei pound kaa devaluation ho.

The subjunctive 'ho' here indicates the ' actual' possibility of the pound being devalued.

Once we found a relationship between the 'potential' of root and the ' theoretical' of epistemic, now we will see if there could be an equivalent to the ' theoretical/factual' distinction of epistemic in the domain of root modality as well. The domain of root modality in Hindi, which has very thinly sliced categories , present a neat division on this line as well. Here we use the terms 'unactualised/ actualized' with respect to the ability's having been realized or unrealized. Unlike the 'factual' in epistemic possibility, the actualized in the root modality is, however, represented by another modal auxiliary. Let us have a look at this division and also its corresponding division in the epistemic, side by side.



Apart from 'le'(the auxiliary for actualized event in the root)there is another marker 'paa' for actualized event used in the negative or interrogative sentences. However, along with its characteristics as actualized marker, 'paa' also has many other modal shades which are not our direct concern here, and hence, here we will not go into its detail any more.

In Bangla, the situation is slightly different. It does not have a modal apparatus for showing such divisions between potential /actualized or potential/ factual. Bangla has only one modal auxiliary 'paar'used in all the contexts. However, unlike Hindi 'sak',which denotes only theoretical or potential , Bangla 'paar' also shows actualized event. The question is, then, how does it differentiate between potential and actual? Let us have a look at the contrast between Hindi 'sak' and Bangla 'paar':

4. kya aap roz saat baje pahunch sakte hein?

(can you reach at seven o'clock every day?) Unactualized event, i.e., futurity is indicated.

5.kya aap roz saat baje pahunch paate hein?
(Are you able to reach at seven o'clock every day?)
Actualized event.

Bangla employs the same modal auxiliary – 'paar', in both the contexts. However, for the unactualized event, it will take the future tense marker. Thus the Bangla equivalents for 4. and 5. will be 6. and 7. respectively.

6.aapni ki roj shomaye pouchate paarben?
7. aapni ki roj shomaye pounchate paaren?

In the domain of epistemic modality again ,for denoting 'factual' possibility , it will take the adverb 'perhaps'along with the future tense form of the verb.Thus , the Bangla equivalents of Hindi epistemic theoretical and epistemic factual (given in the diagram on page 4) will be 8. and 9. respectively.
8.she aksho metre douRte pare.
9.she hoyto aksho metre douRobe.

The difference between 'theoretical' and 'factual' possibility actually lies in the time-reference being non-specified and specified respectively. This is also exhibited by 'paar' which , though a modal auxiliary, behaves like a finite verb with respect to verb conjugation.

3. Conclusion

From the above analysis , thus , one can be convinced that the ambiguity between root and epistemic modality found in many languages is not just an accidental fact but there is underlying semantic affinity between the two. The results of the analysis also attests some generalization regarding the nature of a language with respect to the dominance of tense, aspect and mood (Bhatt, D.N.S.1999). The data from Hindi and Bangla confirm the fact that Hindi is a mood-dominant whereas Bangla is a tense-dominant language.

References

[1] Bhat, D.N.S.(1999) The Prominence of Tense, Aspect and Mood. John Benjamins
[2] Leech, G.N (1971) Meaning and the Englisg Verb. Longman.
[3] Niels Davidsen- Nielsen (1990) Tense and Mood in English: A Comparison With Danish. Mouton
[4] Perkins, Michael,R.(1983) Modal Expressions in English . ABLEX Publishing Corporation
[5] Steele, S (1975) Is It Possible? Stanford University Working Papers on Language Universals 18, 35--58

MACHINE TRANSLATION SYSTEM

S.Mohanty
Dept. Computer Science & Application
Utkal University
Bhubaneswar
ORISSA
Email: sangham1@rediffmail.com

R.C.Balabantaray
The ICFAI Institute of Science & Technology
Fortune Tower, Chandasekharpur
Bhubaneswar
ORISSA
Email: rakesh_b_ray@rediffmail.com

Abstract:

Despite considerable investment over past 50 years, only a small number of language pairs is covered by MT systems designed for information access, even fewer are capable of quality translation or speech translation. This paper presents an approach for machine translation based on syntax and semantics of language to open the door towards MT of adequate quality for English to Indian languages.

One of the greatest challenges before the Information Technologists is to overcome language barriers across the whole humanity so that anyone could talk and communicate to anyone else on the planet in real time. Furthermore, the task of bridging the digital divide can never be accomplished in real sense without breaking the language barrier with the machines. In India we have 18 major languages written in ten different major scripts. In addition we have hundreds of living dialects being used in a predominant manner in different parts of the country. Technological support in the form of machines aids for translation is of immense importance to the country. The dream of building machines that let people from different cultures talk to each other easily is one of the most appealing of Natural Language Processing (NLP) researchers. But Machine Translation (MT) is a difficult problem[4]. The language contains infinite number of sentences and words. Many words have several meaning and sentence can have different meaning in different context. The sentence is the basic language elements, which is a combination of meaningful words to express a complete thought. A sentence must have a subject and a predicate. The subject is the backbone of a sentence and the predicate gives life to the subject. A word functions in a sentence as a part-of-speech such as noun, verb, adjective, adverb and preposition. A language processing system must have considerable knowledge about the structure of the language including what the words are and how they combine into phrases and sentences. It must also know the meaning of the words and how they contribute to the meaning of a sentence and to the context within which they are being used. To determine the meaning of a sentence, we first determine the meaning of the individual words. India is highly multilingual country with a large number of living languages. The beauty of Indian languages is that they are verb ending and the word group order is free to be placed at any place within the subject and verb with lots of structural similarities. Indian languages can be classified into four broad groups according to their origin. These are:

- Indo-Aryan family (Hindi, Bangla, Asamiya, Punjabi, Oriya, Gujurathi etc.)
- Dravidian family (Tamil, Telgu, Kannada, Malayalam)
- Austro-Asian family
- Tibetan-Burmese family

Within each group the languages exhibit a high degree of structural homogeneity. We have exploited this similarity upto a greater extent in our system.

The architecture of our System is divided into six parts such as Parser, Translator, OMT System, OMT database, Disambiguator and the Software tools used to see the result (figure 1).

The heart of the OMT system is the OMT database (bilingual dictionary). In this database we have stored various information such as: English word, category, tense, Oriya meaning of the word etc. The Oriya meaning of the words has been stored in simple ISCII format. The system is developed using Java programming language and MySQL as the database.

The parser takes the English sentence as input from the OMT system and then each and every sentence is parsed according to various rules of parsing with the help of OMT database. During parsing in some cases it is taking the help of morphological analyzer (Ex: for words like boys morphological analyzer will convert it to boy). The translator takes the parsed structure of sentences from the OMT system as input and then it will perform the task of translation with the help of OMT database. This translator part is taking the help of tense analysis and verb tagging module. Word sense disambiguation is a major problem in MT. This dis-ambiguator module is doing the task of disambiguation with the help of frequencies obtained from Corpora (Oriya Corpora) by using the n-gram model [10].

We have implemented the OMT system for various types of simple as well as complex

sentences. It has been designed in a fully extensible manner (i.e. we can add/modify new rules for translation with the existing system). In the first phase the system is tested with various types of sentences taken from schoolbooks. One of the salient features of our system is that, it is capable of translating the scanned files from the books as well as newspapers. For word sense disambiguation, the testing has been made for various words like: plant, bank, tank etc. It is showing pretty accurate results according to the contexts. In the translation browser, the standard X-windows mouse functions are used to open, close, move the window and change its size. The translation browser provides the facility that one can enter the text for translation in the browser or he can get the input text from the previously stored file by using a particular button.

This work is part of an ongoing work on more realistic OMT system, which is totally based on object-oriented paradigm. The structural parsing technique of our OMT system can be used for the translation of English to any other Indian language, as they are almost similar in their structure, being originated from Sanskrit and influenced by Prakrit and Pali. We are working on to incorporate the pragmatic analysis to our system in the next phase.

References:

[1]. Allen. J. (1987) " Natural Language Understanding". The Benjamin / Cummings publishing company, Inc Menlo Park.

[2] Bharati. A. et. Al. (1995), " Natural Language Processing, A Paninian Perspective". Prentice Hall of India Private Ltd., New Delhi.

[3]. Kar Pandit K.C (2000), "TaruNa shabdkoSh", Grantha Mandira, Cuttack.

[4]. Manning C. and Schutze H. , " Foundation of Statistical Natural Language Processing", The MIT Press, Cambridge, Massachusetts, London, 2001.

[5]. Mohanty S. et. al. , " Word Sense Disambiguation for OMT", National Conference on AIETFA, Bhubaneswar, Orissa.

[6]. Mohapatra Pandit N. and dash S. (2000), " Sarbasara Byakarana", New Students Store, cuttack, Orissa.

[7]. Russell and Norvig, " Artificial Intelligence A Modern Approach", Pearson Education, India, 2001.

[8]. Yarowsky D.(1993), "One Sense Per Collocation", Proceedings of ARPA Human Language Technology Workshop, Princeton.

[9]. Yarowsky D.(1994), " Decision list for Lexical ambiguity resolution : Application to Accent restoration in Spanish & French", Proceedings of the 32nd Annual Meeting of the ACL, Las Cruce, NM.

[10] Yarowsky D.(1995), "Unsupervised word sense disambiguation rivaling supervised methods", Proceedings of the 33rd Annual meeting of the ACL,PP 189-196.

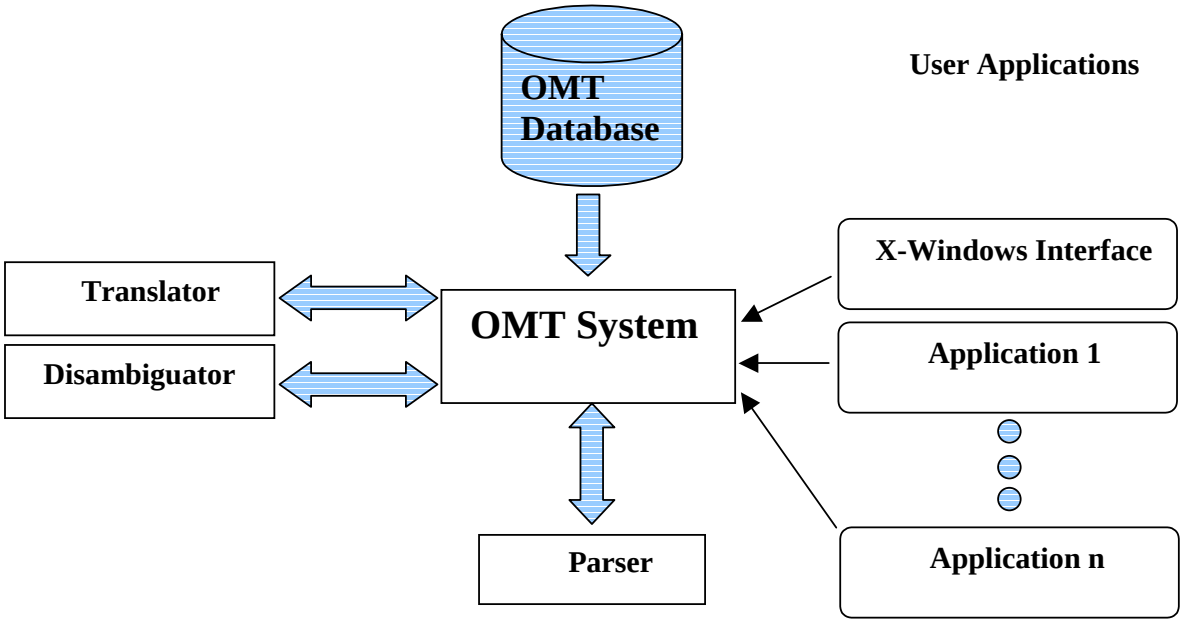


Figure 1: Architecture of the System

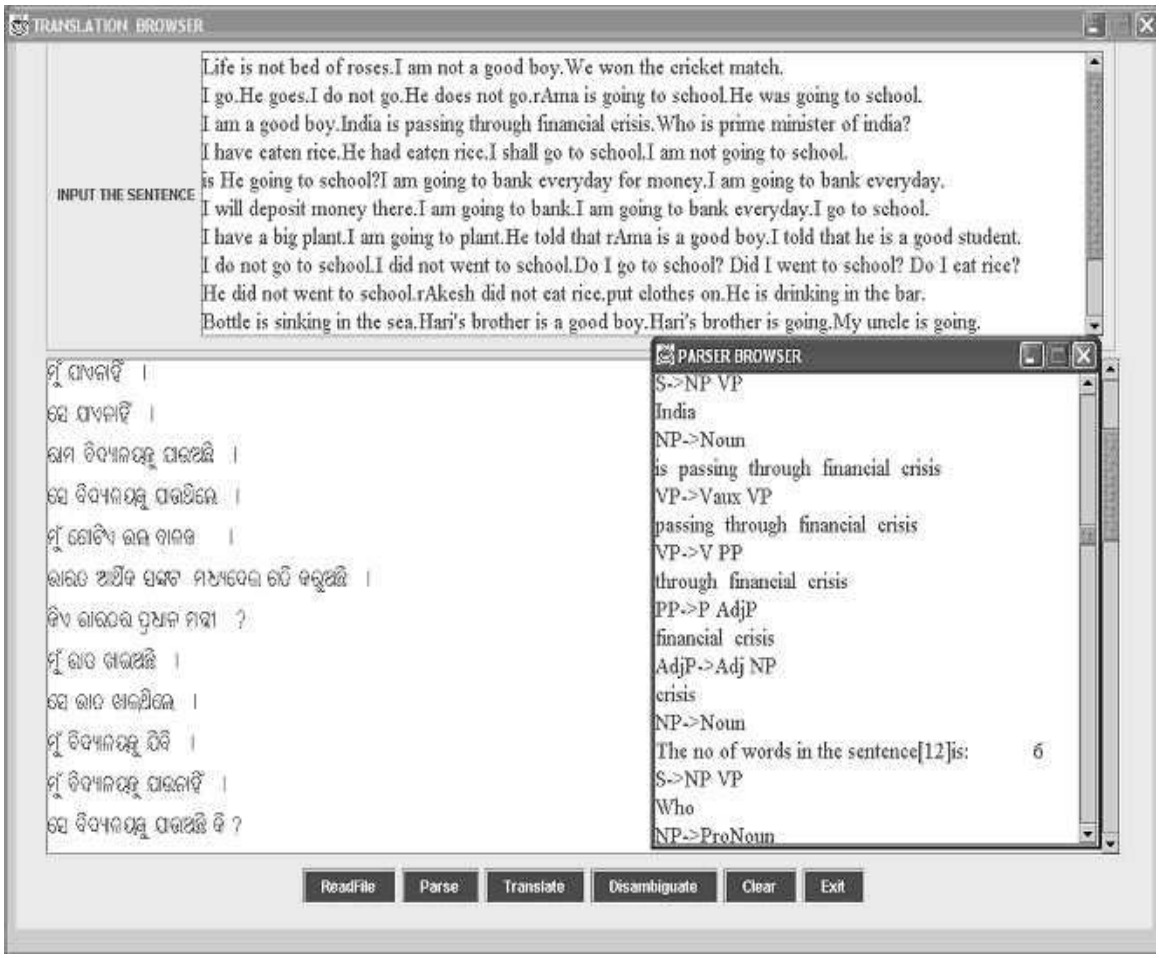


Figure 2: Output of the OMT System

Universal Networking Language Based Analysis and Generation Of Finite and Non-Finite Verbs In Bengali

Kuntal Dey and Pushpak Bhattacharyya

February 19, 2004

Kuntal Dey: u2ckuntal@yahoo.com
Pushpak Bhattacharyya: pb@cse.iitb.ac.in
Computer Science and Engineering Department,
Indian Institute of Technology, Bombay, India.

Abstract– *Verb forms the backbone of any natural language. Hence, proper processing of verb is a primary requirement for processing any natural language. In this paper, we present the computational analysis of the verb structure in Bengali - a member of the Indo Aryan family of languages - with a view to interlingua-based machine translation. Bengali is ranked 4^{th} in the list of languages ordered according to the size of the population that speaks the language. Extremely interesting language phenomena involving morphology, case structure, word order and word senses make the processing of Bengali a worthwhile and challenging proposition. In the present work we have focused our attention on Bengali verb analysis and corresponding generation. A recently proposed scheme called the Universal Networking Language (UNL) has been used for this purpose. The approach is adaptable to other members of the vast Indo Aryan language family. The parallel development of both the analyzer and the generator system leads to an insightful intra-system verification. Our approach is rule based and makes use of authoritative treatises on Bengali grammar.*

1 Introduction

Bengali is spoken by about 189 million people and is ranked 4^{th} in the world in terms of the number of people speaking the language ([2]). Like most languages in the Indo Aryan family, descended from Sanskrit, Bengali has the SOV structure with some typical characteristics. Creating a system for processing Bengali creates the possibility of laying the framework for processing many other Indian languages too.

Work on Indian language processing abounds. *Project Anubaad* for machine translation from English to Bengali developed in Jadavpur University uses the *direct translation approach*. *Angalabharati* [16] system for English Hindi machine translation is based on pattern directed rules for English. In MATRA [10], a web based MT system for English to Hindi in the newspaper domain. Among other well-known MT systems and projects are the *MANTRA* MT system [1], Project *Anusaaraka* [6] *etc.* References to most of these works can also be found at [3]. Other famous MT systems are *Pivot [15], Geta [9], SysTran [13]* etc.

The *Universal Networking Language (UNL)* has been defined as a digital meta language to describe, summarize, refine, store and disseminate information in a machine independent and human language neutral form. The information in a document is represented sentence by sentence. Each sentence is converted into a directed acyclic hyper graph having concepts as nodes and relations as arcs ([4]). Knowledge within a document is expressed in three dimensions:

1. Word Knowledge is expressed by Universal Words (UWs) which are language independent. UWs are tagged using restrictions describing the sense of the word in the current context. For example, *drink*($icl >$ *liquor*) denotes the noun sense of *drink* restricting the sense to a type of *liquor*. Here, *icl* stands for inclusion and forms an *is-a* relationship like in semantic nets [7].
2. Conceptual Knowledge is captured by relating UWs through a set of UNL relations [12]. For example, *Humans affect the environment* in UNL is:

   ```
   agt(affect(icl>do).@present.@entry,
   human(icl>animal).@pl)
   obj(affect(icl>do).@present.@entry,
   environment(icl>abstract thing).@pl)
   ```

   Here *agt* is the *agent* and *obj* the *object*. $affect(icl >$ *do*), $human(icl >$ *animal*) and $environment(icl >$ *abstract thing*) are the UWs denoting concepts.
3. Speaker's view, aspect, time of event, etc. are captured by UNL attributes. For instance, in the above example, the attribute *@entry* denotes the main predicate of the sentence, *@present* the present tense and *@pl* the plural number.

The detailed specification of UNL can be found at [5]. Our work on Bengali verbs is based on an authoritative treatise on Bengali grammar [8]. The strategies of analysis and generation of linguistic phenomena have been guided by rigorous grammatical principles.

2 EnConverter and DeConverter machines

The EnConverter (*EnCo*) [18] is a language-independent parser, a multi-headed Turing machine [11] providing a framework for morphological, syntactic and semantic analysis synchronously using the UW dictionary and analysis rules. Analyzing the natural language input, it generates semantic relations between the UWs and/or attaches speech act attributes to them. The final output is a set of UNL expressions equivalent to a UNL graph.
The DeConverter (*DeCo*) [17] is a language-independent generator that produces sentences from UNL graphs. Like EnCo, DeCo too is a multi-headed Turing Machine. It does syntactic and morphological generation synchronously using the lexicon and the set of generation rules.
In order to adapt the UNL engines to enconvert the Bengali sentences into the UNL interlingua/graph and to deconvert the UNL interlingua/graph into Bengali sentences, an enconverter rule-base and a deconverter rule-base have been written. For an exposure of the working of rule theory, [14] can be referred to. The rules within the rule-base are compliant with the corresponding UNL engines and are focused to deal with the Bengali language structure. A subset of these rules take care of the language phenomena associated with the verbs.

3 Verb in Bengali: *Kriyaa*

Verb performs the action stated by the sentence. In Bengali, verbs are present directly, and the verb is known as *kriyaa* (ক্রিয়া). In the Indian linguistic system - descended from Sanskrit - the verb-related phenomena in Bengali gives rise to interesting variations of scenarios. The *kriyaa*s (verbs in Bengali) are classified into 2 types [8]. They are *samaapikaa* (সমাপিকা) and *asamaapikaa* (অসমাপিকা). *Samaapikaa kriyaa* corresponds to the finite verb and *asamaapikaa kriyaa* corresponds to the non-finitie verb (infinitive) when compared to the classical definitions of the sub-categories of verbs. Each of them have their own subdivisions. They have been exemplified later within this section. The relations in the UNL that are covered by the verbs section and other types of sentences having interesting verb-related phenomena are mainly *con, seq, man, via, rsn etc.* There are also quite a few attributes covered. The most important one is the @entry attribute mentioned earlier. Among the others, *@should, @ability, @progress, @complete, @insistence* and many other attributes can result from variations on the verbs. Attributes such as *@progress, @complete* etc are generated from the tense of the verb. A separate work has been carried out to cover the classical tenses in Bengali, which is outside the scope of this paper. It is worthwhile to mention that along with these relations generated in the UNL by various verb phenomena, the critically important *kaarak* theory has also been handled in the project. The UNL relations covered there include *agt, cag, ptn, aoj, cao, obj, ben, cob, ins, met, gol, pur, rsn, frm, src, plf, tmf, mod, pos, pof, plc, plt, tim, tmt, opl, scn, to, etc.* The mapping from language phenomena to UNL relations is deterministic in nature, and although it is not one-to-one (for example, the *rsn* relation can be generated both by *kaarak*s and verbs, or say, more than one type of *kaarak* can generate the same UNL relation), but the mapping is not very divergent either; rules can be formulated to resolve during generation the mapping obtained during analysis appropriately to generate the correct output. An exhaustive study of *kriyaa* with a target to analyze Bengali into UNL has been carried out. The foundation of this work is the *kriyaa* system for Bengali as found in [8].

3.1 *Samaapikaa kriyaa*: The finite verb

Samaapikaa kriyaa (সমাপিকা ক্রিয়া) corresponds to the finite verb, and is capable of completing the sense of action stated by the sentence without the requirement of presence of any other verb. *Samaapikaa* verbs can be categorized into two:

1. *Samaapikaa sakarmika*: The *sakarmika* (transitive) verb has an object associated with it, and hence an *obj* or equivalent relation is generated in the UNL graph. Example: *se raamke cithi likhbe gaacher chaayay base* (He will write a letter to Ram sitting in the shade of tree). A relation *obj(write:0F.@entry.@pred.@future, letter:0A)* will be generated.
2. *Samaapikaa akarmika*: The *akarmika* (intransitive) verb does not have an object associated with it, and hence an *obj* or equivalent relation is not generated. Intransitive verb phenomena is compulsorily within this class. Example: *shishuraa khelchilo maayer kole.* There is no *obj* or *ben* or equivalent relation generated by analysis.

3.2 *Asamaapikaa kriyaa*: The verb infinitive

Asamaapikaa kriyaa (অসমাপিকা ক্রিয়া) is a verb that can't complete the sense (of action) of a sentence on its own, and there is at least one finite verb (*samaapikaa kriyaa*) in the sentence. *Asamaapikaa* verbs can be categorized into three:

1. Those formed by *verb*+ইয়া (*iya*). The relations generated from this category of অসমাপিকা verbs are primarily *seq, man, via, rsn etc.* Example of generation of *seq* relation for Bengali and the resulting UNL graph folows:
Input to enco: কলেজ থেকে ফিরে খেলতে যাই
kalej theke phire khelte jaai.
College-from return-after play-for go-I.
I go to play after returning from college.
Output of deco: aami kalej theke phire khelte jaai
Equivalent: আমি কলেজ থেকে ফিরে খেলতে যাই
Strategy of analysis:

- Since the first verb (*phire*) is an EVENT verb having an *e* (এ) *bibhakti* and the second verb (*jaai*) is the last verb of the sentence, a *seq* relation is resolved between these two.

Salient rule:

- >{V,MORADD,BLKINSERT,eADD,EVENT:-eADD, +SEQRES:seq:}{V,MORADD:+@::}P22;

UNL generated by enco:

agt(go(icl>action):0O.@entry.@pred.@present, I(icl>person):00)

seq(go(icl>action):0O.@entry.@pred.@present, return:0C.@present)

gol(go(icl>action):0O.@entry.@pred.@present, play(icl>event):0H.@past)

src(return:0C.@present, college:00)

Remark: An interesting point here is that the agent (*aami*), which is not present in the input sentence of the enconverter explicitly, has been explicitly added in the output sentence of the deconverter.

Another example: The following sentence generates a *man* relation:

Input to enco: নৌকা পাল তুলে যাচ্ছে

noukaa paal tule jaacche.

Boat sail lifted going.

Boat is going with its sail lifted.

Output of deco: noukaa paal tule jaacche

Equivalent: নৌকা পাল তুলে যাচ্ছে

Strategy of analysis:

- Phrase tag is inserted. Sentence becomes: *noukaa paal tule jaacche.*
- Compound UW is generated for the portion of sentence contained inside the phrase tag.
- This compound UW is resolved as the child of a *man* relation in the UNL graph.
- After appropriate handlings of the phrase tag, there is no trace of the phrase tag, only a flag information denoting *phrase_committed* remains.
- Now a *man* relation is resolved, guided by this *phrase_committed* flag and the presence of the two verbs in sequence, of which the first one also has an *e* (এ) *bibhakti.*

Salient rule:

- >{V,eADD,phrase_committed:+&@entry:man:}{V:+MANRES::}(STAIL)P22;

UNL generated by enco:

agt(go:0O.@entry.@present.@progress.@pred, boat(icl>transport):00)

man(go:0O.@entry.@present.@progress.@pred, :01)

obj:01(lift(icl>do):0E.@entry.@pred.@present, sail(icl>thing):09)

2. Those formed by *verb*+ইলে (*le*). Mainly results in generation of a condition (*con*) relation. Example:

ভালো লাগলে আবার নিতে পারি

bhaalo laagle aabaar nite paari.

Good feel-if again take-can.

If I feel good I can take again.

Output of deco: jadi bhaalo laage tabe aami aabaar grahan karte paari

Equivalent: যদি ভালো লাগে তবে আমি আবার গ্রহণ করতে পারি

3. Those formed by *verb*+ইতে (*te*). Relations such as *rsn* and various attributes such as *@progress, @insistence, @ability, @should etc* have been found to be generated from this kind of অসমাপিকা verbs. An example follows:

Input to enco: রাম ক্রিকেট ভালো খেলতে পারে

raam kriket bhaalo khelte paare.

Ram cricket good play-can.

Ram can play good cricket.

Output of deco: raam bhaalo kriket khelte paare

Equivalent: রাম ভালো ক্রিকেট খেলতে পারে

Strategy of analysis:

- The word *paare* (পারে) is a *case* that denotes ability. Hence, seeing it, an *@ability* attribute is generated.

Salient rule:

- +{V:+pAreADD,+&@ability,+CASEADD,-&@future,-&@past,+&@present, +3PERSON,-2PERSON,-1PERSON::}{[[*pAre*]],CASE:::}P30;

UNL generated by enco:

agt(play(icl>event):0I.@entry.@ability.@present, Ram(icl>person):00)

obj(play(icl>event):0I.@entry.@ability.@present, Cricket(icl>game):05)

man(play(icl>event):0I.@entry.@ability.@present, good(icl>characteristic):0D)

4 Conclusion

Complete processing of verb is indispensible for natural language processing system. In this paper, we have outlined a system for computational analysis of the Bengali verb for UNL-based MT. The parallel implementation of the analysis and generator system provides the platform for intra system verification. Inhouse cross system verification is done using the Hindi system (also under development). Apart from systematic approach to processing of verbs, addressing complex phenomena involving adjectives and adverbs is also being looked into.

References

[1] *http://www.cdacindia.com/html/about/success/mantra.asp.*

[2] *http://www.harpercollege.edu/~mhealy/g101ilec/intro/clt/cltclt/top100.html.*

[3] *http://www.tdil.mit.gov.in/mat/ach-mat.htm.*

[4] *http://www.unl.ias.unu.edu.*

[5] *http://www.unl.ias.unu.edu/unlsys.*

[6] Bharati A., Chaitanya V., and Sanyal R. *Natural Language Processing: A Paninian Perspective.* Prentice Hall India Private Limited, Oct 1996.

[7] Woods W. A. *What's in a link: Foundations for semantic networks.* Morgan Kaufmann Publishers, Inc.

[8] Chakrabarti B. *Uchchatara Bangla Byakaran.* Akshay Malancha, Oct 1963.

[9] Vauquois B. and Boitet C. Automated translation at grenoble university. *acl.ldc.upenn.edu/J/J85/J85-1003.pdf <http://acl.ldc.upenn.edu/J/J85/J85-1003.pdf>.*

[10] Rao D., Mohanraj K., Hedge J., Mehta V., and Mahadane P. *A Practical Framework for Syntactic Transfer of Compound-Complex Sentences for English-Hindi Machine Translation.* KBCS, Mumbai, 2000.

[11] Hopcroft J. E. and Ullman J. D. *Introduction to Automata Theory, Languages and Computation.* Addision-Wiseley Publishing Company, 1979.

[12] UNL Centre/UNDL Foundation. *The Universal Networking Language (UNL) Specifications.* November 2001.

[13] Hutchins W. J. and Somers H. L. *An introduction to Machine Translation.* London: Acamedic Press, 1992.

[14] Dey K. and Bhattacharyya P. *Universal Networking Language Based Analysis and Generation of Bengali Case Structure Constructs.* December 2003.

[15] Muraki K. *PIVOT: Two-Phase Machine Translation System.* Hanakone, Japan, 1987.

[16] Sinha R. M. K. *Machine Translation: The Indian Context.* International Conference on Applications of Information Technology in South Asian Languages, 1994.

[17] UNU/IAS. *DeConverter Specifications.* UNU/IAS UNL Center, Nov 1998.

[18] UNU/IAS. *EnConverter.* UNL Centre/UNDL Foundation, May 2001.

NLP APPLICATIONS

A Machine Translation and Natural Language Generation Framework using a semantic frame based interlingua

Sudeshna Sarkar, Anupam Basu, Pradipta Ray, Monojit Choudhury, Samit Bhattacharyya
Computer Science & Engineering
Indian Institute of Technology, Kharagpur : 721302
Email: {sudeshna,anupam,pradipta,monojit,samit}@cse.iitkgp.ernet.in

***Abstract*— In this report we describe work related to Machine Translation being carried out at IIT Kharagpur. We are aiming at an interlingua based machine translation framework. But our design is being specifically fitted to capture translation to and from Indian languages like Bengali and Hindi. We are also considering translation to and from English. In this paper we describe the major components of the system including the source analyzer, interlingua representation and the generation module. We also describe an iconic communication system which takes as input a set of icons and generates sentences in a natural language.**

I. Introduction

In this work we describe our generic framework for language analysis and generation. The input may come from a source language which may be a natural language X, or can be a set of icons where each icon represents one concept, action or a property. The output will be a sentence in the target language Y. We use an interlingua to represent a sentence internally based on a semantic frame based representation. The representation is largely language independent. However we use the lexicon items of the given languages to minimize loss during bilingual transfer.

The system consists of the following modules :

1) Source language analyzer : This takes a text in the source language X and translates it to interlingua with lexicon X At this stage clause separation and role assignment are done.
2) Bilingual word and phrase dictionary are used to translate interlingua in lexicon X to interlingua in lexicon Y. Word sense disambiguation must be performed in this step to identify the correct lexical item in Y for a given lexical entry in X
3) Target language generator takes interlingua with lexicon Y and generates the surface form of Y

The system is designed to use the following knowledge sources : Bilingual word dictionary, phrase dictionary and dictionary of idioms. The following additional resources may be used to reduce error in translation due to error in role assignments: argument structure of different verbs, linguistic data about the preferred ontology of objects that map to important roles of given verbs, and statistical data about co-occurrences of word or concept pairs.

II. Frames Representation

While designing the intermediate representation format we have kept in mind the language constructs of English, Bengali and Hindi. The first phase of the design is tuned to a controlled vocabulary and restricted grammatical constructs.

A sentence is represented by a set of frames (*FR*s). Each sentence is associated with a primary *FR*. An *FR* has a *main verb*, and a set of *role*s (or concepts). The *main verb* is represented by the *root*, and its *tense* (present, past or future). For an *FR* which represents a subordinate clause the *main verb* may be in other forms like *participle* form (*asamApikA kriYA*), *infinitive* form, etc. In such cases we indicate this in place of *tense*. We also indicate the *aspect* (simple, continuous or perfect) and *modality* of the verb. The prominent modalities are *normal (default), possibility, ability, likelihood, permission, unacceptability, instruction, request, suggestion, intention, invitation, offer, unwillingness, refusal, wish, importance*, etc.

Every frame has a tag indicating its type. The type of a frame may be *declarative, yes-no-interrogative, wh-interrogative, imperative or negative*.

A role has an associated *type*. Some of the common role types are enumerated below. A role may have property/ies which are represented by qualifiers associated with the role. The frame entry consists of a headword and optionally adjective(s) or qualifier(s). The headword can be another *FR* (these represent nominal clauses), or it can be a noun, pronoun, adverb etc. Qualifiers can represent single words like an adjective or a determiner, an adjective phrase, or an adjective clause.

Being able to identify the roles correctly is helpful in synthesizing the corresponding object to a correct target language surface representation. We are taking into account the common role distinctions needed in English, Bengali and Hindi to bring out the semantics of the objects and for being able to generate the correct inflections of the words and inserting appropriate postpositions. The core case roles that are part of the valency of a verb are :

- agent : animate instigator of an action
  (Ram) broke the window.
- experiencer : animate cogniser
  (Rabi) heard Vimala come. (Tina) felt happy.
- patient : the animate entity affected by the action or event
  Manas kicked (Bikas). (Mira) fell over.

- theme (patient) : entity in or undergoing a change
  Anju kicked (the ball).
- recipient : of a transfer
  Srija got (Titir) a present.
- force / cause : non-cognizant origin of an event / the force bringing about a change of state.

The oblique cases can be attached with most verbs and hence are not a part of the valency of the verb. The important oblique case tags are

- time : <point, interval, bound> at which a situation holds
  I went to Delhi (yesterday).
- location/ place : at which a situation holds
  Shankha plays football (in the park).
- instrument : tool (inanimate accessory) used by an agent
  I dug the hole (with a spade).
- beneficiary : for whose benefit an action is taken
  Ram bought flowers (for Sita).
- Comitative (also called Accompaniment)
  I had dinner (with Pradipta).
- reason : for which an action is undertaken
  I went to the park (to play).
- manner
  He speaks (like a tornado).
- stimulus : of a mental / perceptual state or event
- source : from which movement is directed
  Mr Roy arrived (from Mumbai).
- goal : toward which movement is directed
  Ram went (home). Rabi brought some books (to Soma).
- trajectory : motion from source to destination takes place over the trajectory
  Sital drove to Chennai (along NH5).

There is no consensus among the researchers about how many and which roles to use. Some people recommend using only 3-4 roles, others advocate using a few hundred. For example, in the FrameNet [1] project many roles have been used which are often very specific to a particular word. A related problem with semantic roles is "role fragmentation", that is, how finely thematic roles should be divided.

Our approach is to start with a few most common roles. We will then look at the problem of role assignment in Bengali, Hindi, and English and find out the error rate in automatic role assignment. We will also like to evaluate the suitability of these roles in being able to generate a correct translation in the target language with a minimal loss in information. We prefer to have a hierarchical structure among roles, so that in some cases where we fail at a finer level of granularity we will succeed at a higher level. We will test whether this will still enable us to generate a correct translation for several sentences in the target language. For example, we may club agent and experiencer into one higher level role, patient and theme into another higher level role. The time role may be subdivided into several finer roles (at-time, from-time, to-time). Each of these may again have subtle variations.

Simple sentences which have only one clause are represented by a single *FR*. Sentences involving multiple clauses are represented by multiple *FR*s. Compound sentences consist of more than one *FR* and the relation between them : the relation can be conjuncts like and, or, but; or temporal (before, after, during), spatial relations (in front of, beside, behind, above, etc,), causal relations (because), conditional relations (if-then, if-the-else) etc. In complex sentences, a subordinate clause can be a noun clause, or adverb clause and they will be associated with a role. An adjective clause will be associated as a qualifier of an object which represents one role.

III. Source language analysis

Arguments are the constituents that are required for a sentence to be grammatical (the subject, the object, or the oblique of a sentence). Argument structure represents the thematic roles a verb has, and is characterized as the interface level between the lexical semantics of a verb and its syntax. We may divide the verbs into groups on the basis of their argument structure.

The objective of the source analyzer is to analyze a single sentence in source language X to its frame representation. We have decomposed the problem into the following parts :

- Morphological analysis of all the words : We have implemented a morphological analyzer for Bengali and Hindi that fully handles inflectional morphology. We also have made some progress in handling derivational morphology.
- Part of speech tagging : We have currently used a small tagged corpus to train the system. We plan to tune the system by selecting useful training examples to improve the accuracy of the system.
- Local word grouper : We have made a rule based local word grouper for Bengali and Hindi.
- Clausal analyzer : The clausal analyzer takes a sentence and finds the constituent clauses of the sentence.
- Role assignment : This involves taking a word group or constituent in a sentence and a target frame, and automatically labeling the constituent with a semantic role [3]. For role semantics to become relevant we need robust methods for automatic role assignment.

We will illustrate the complexities of some of these tasks with examples from the Bengali language.

A. Role Assignment

We use a robust mapping of the output of the parser to the frame based representation. This means that every word group must be assigned some roles. In situations where we are not able to pinpoint the correct role, we may represent the role as a disjunction of role types (because we may be able to eliminate certain role assignments easily).

One approach is to individually rank the possible role assignments of an object on the basis of world knowledge (available from domain knowledge and ontology) and statistical information gleaned from corpora. We may also use constraint satisfaction techniques to eliminate

some possibile role assignments, and generate possible sets of unique role assignments. The probabilities of these various interpretations can be computed on the basis of co-occurrence probabilities that we can estimate from a corpus. This will help us select the most likely interpretation.

Role assignment is a non-trivial problem in some Indian languages which are relatively free word order. In Sanskrit and many modern Indian languages the *vibhakti*s and the post-positions help us to identify the possible roles. However in languages like Bengali *vibhakti*s can rarely be used to uniquely determine the roles. The problem is that in many cases we have *shUnya vibhakti*. In Bengali there are only 5 *vibhakti*s *(e, ke, te (ete), ra (era), re)*, out of which *re* is used only in poetry and *ra (era)* is used only for possessive. The remaining three *vibhakti*s and the *shUnya vibhakti* are used for all roles. The postpositions (*anusarga*) also help in determining the roles.

In Bengali we find that many roles can be expressed by a number of *vibhakti*s, and that most *vibhakti*s can be associated with multiple roles. We conclude that syntactically it is impossible to do correct role assignment. Accurate role assignment requires semantic knowledge about what types of objects are preferred for which roles. We advocate the use of a powerful ontology to support the role structure. A good knowledge base containing the possible ontology categories that can take on specific roles for different classes of verbs will help a great deal in accurate role assignment.

Role assignment can be modelled as a classification task. In existing work [4] the following features have been used : syntactic information (path from predicate to constituent, phrasal type of constituent) and lexical information (head word of the constituent, predicate). In order to generalize from examples, the following can play a key role : grouping of verbs according to their argument structure, and the ontology of the objects.

B. Clausal Analyzer

The clausal analyzer does the following:

1) Decompose the sentence or any other form of input into a set of clauses. Mark the primary clause.
2) Identify the relationships between the clauses in 1.

Every clause is represented as a semantic frame. The *Clausal analyzer* makes certain assumptions:

1) Every clause has a *samApikA kriYA* (or main verb).
2) Sometimes clause markers like who, what, which, whom (*ye*[1], *ye ... se, yakhana, yakhana ... takhana, yeTA, yAke* etc. in Bengali) can be used as clause markers.

However these conditions are often violated. In Bengali, a *samApikA kriYA* (typically forms of to be, to have) are often implicit. (Example : *tAra manaTA besha bhAlo*), Clause markers are often absent. (Example : *mane pa.De paha.De uThechhilAma, bhebe dekhalAma, bA.DI jete chAi*) We are trying to build a clausal analyzer for Indian languages like Bengali and Hindi. Every clause needs to have a *main verb*. After identifying the clauses, we may need to insert verbs where they are implicit. We also will like each clause to have an *agent* or an *experiencer*. Where these roles are absent, they may have to be deduced from the form of the verb. In a complex or compound sentence, the subject is sometimes mentioned once and shared between clauses. Such transfer rules need to be found.

[1]All Bengali language examples are written using itrans

C. Handling of secondary verbs

A simple sentence has one main verb, but many natural language sentences contain other secondary verbs. Auxiliary verbs like want, request can be translated into the modality of the other verb in the clause denoting the event. Some other examples are the use of participles and infinitives in English and of *asamApikA kriYA* in Bengali. In a sentence where an *asamApikA kriYA* occurs, it can be part of an adjective phrase, it can denote a separate frame which occurs before or simultaneously with the event of the main verb. It can denote a frame which has a causal or conditional relationship with the frame of the main verb. It may denote repeated action. It can be part of a role of type reason, of type theme, of type manner, etc. If a separate frame is indicated we can represent the resulting structure as we would do a compound or complex sentence. Otherwise it will be a part of a role.

D. Transfer from interlingua-X to interlingua-Y

This will require a bilibgual dictionary and word sense disambiguation. Word sense disambiguation will be done on the basis of selection preference between two grammatical relations (subject-verb, verb-object, head-modifier). This helps in the accurate choice of the target language word (lexical choice) given multiple senses.

IV. Iconic interface for input

We have developed an iconic communication system which supports any fixed vocabulary of icons and a set of natural languages. The users can select a set of icons through the interface. The objective of the system is to generate a plausible sentence in one or more of the languages. Each icon corresponds to a unique lexical item in multiple languages. In the first version of the system, the user explicitly assigns each object selected by one or more icons to a unique role. In this system the user in effect fills up the interlingua. The problem is only of generation. In the next version of the system, our objective is to let the user select a set of icons including at least on verb. The system will have the goal of coming up with a plausible sentence containing the vocabulary items selected. The problem that we need to solve here is the “role assignment” problem given no syntactic clue.

An approach to solving this has been discussed in the previous section.

V. Target Language Generation

Given a set of *FR*s representing a particular sentence, our next task is to generate the surface form in target language Y.

A. Generation of simple sentences

For generation in a target language Y the following informations are required :

- Morphological synthesis of individual words: Every individual word's surface form is generated from the root on the basis of pertinent attributes of that word. (eg. in Bengali, the surface form of a verb depends on its number, person, tense, aspect and modality) A table lookup will suffice for this purpose, but we are merely storing the affixes and rules for combining the root and the affix.We have developed a complete morphological synthesizer for Hindi and Bengali using table lookup and finite state methods [2].
- Synthesis of phrases: Every local word group is generated next using the morphological synthesizer for individual words. Such phrase synthesis involves ordering all words in a word group(eg. in Hindi, in a noun group this would correspond to adjectives, qualifiers, nouns, conjunctions and postpositions) around the head word.
- Role ordering: The linear structure of a natural language sentence forces a linear ordering over the occupied roles of a frame during generation. Even free order languages like Bengali and Hindi, while not having a strict linear ordering which must be adhered to, nevertheless have a preferred role ordering.The ordering is usually a function of the frame in question, and which of its roles are occupied.
- Surface form generation: Based on the phrase synthesizer, and the role ordering module, the surface form of the simple sentence is generated

B. Generation of complex and compound sentences

Generation of complex and compound sentences in a target language Y requires the following steps:

- Generation of individual clauses: Individual clauses are synthesized by the simple sentence generator.
- Identification of the relationships between the clauses: The ordering of the clauses and nature of the indeclinable conjunction is determined by the nature of the relationship between the clauses in question. This relation may be a simple conjunction, temporal relationship, spatial relationship, conditional or a causal relationship. A set of rewrite rules may be used for intra-clausal reordering, ordering the clauses and generating appropriate conjunctions based on the nature of the inter-clause relationship.
- Surface form generation of the resulting sentence: Any complex or compound sentence may then be synthesized. For complex sentences, positioning of the clauses will depend on the role assumed by the subordinate clause while for compound sentences, positioning of the clauses will depend on the inter-clausal relationship.

C. Generation with ambiguous role assignment

The source language parser may not be able to unambiguously assign the correct role to every object. To have a robust system, we take every object and find the possible roles the object can take. In case we cannot disambiguate the roles the object is assigned multiple roles. However, even in the face of ambiguous role assignment, the synthesizer may be able to generate a single synthesized surface form which may suffice as the surface form for all the roles in question. (This is especially true in languages belonging to the same family, like Hindi and Bengali, which are both Indo-Aryan languages) Alternately, if confidence measures for each role may be given by the source language parser, then the generator may generate only those surface forms above a certain confidence measure.

D. Robust handling of ambiguities

Another important issue which needs to be resolved during generation is a robust approach in handling incomplete information. If the sentence in the source language is more informative than the target language sentence, the issue is that of collapsing the excess information. (eg. for English to Bengali translation, both "he" and "she" will be mapped to *"se"*) However, a trickier problem occurs when the target language sentences requires more information than the source language sentence can provide. This may be tackled in the following ways:

- The target language sentence may be restructured such that it no longer requires more information than the source language sentence can provide.
- All possible surface forms may be given as disjunctions in the target language sentence. (eg. On encountering a Bengali "se", we may generate "he/she" in English.
- The required information may be gathered by discourse processing of the source or target texts. This is the most satisfactory, but hardest approach.

References

[1] Baker, Collin F., Fillmore, Charles J., and Lowe, John B. *The Berkeley FrameNet project.*, In Proceedings of the COLING-ACL, Montreal, Canada, 1998.

[2] S. Bhattacharya, M Choudhury, A Basu and S Sarkar, *Inflectional Morphology Synthesis for Bengali Noun, Pronoun adn Verb Systems* Tech Report IIT/TR/CSE/2003/AB2, IIT Kharagpur, 2003.

[3] Daniel Gildea and Daniel Jurafsky, *Automatic Labeling of Semantic Roles*, Computational Linguistics, Vol 28.3: 245-288, 2002.

[4] S. Pado and G. Boleda Torrent: *Towards a better understanding of frame element assignment errors.* Proceedings of the Workshop on Prospects and Advances in the Syntax/Semantics Interface, Nancy, 2003.

Multilingual Question Answering

Pushpraj Shukla, Pankaj Goyal, Kumar Kapil, Amitabha Mukerjee, Achla Raina
Indian Institute of Technology, Kanpur
Kanpur – 208016, UP, INDIA
{praj,pankajgo,kapil,amit,achla}@iitk.ac.in

Abstract

Given source text in several languages, can one answer queries in some other language, without translating the sources into the language of the questioner? In this paper we try to address this question as we report our work on a restricted domain, multilingual Question–Answering system, with current implementations for source text in English and questions posed in Hindi or English . The cross-language functionality has been achieved by converting the queries and the documents to an intermediate representation, an inter-lingua called UNL.

1. Introduction

Multilinguality and cross-linguality have emerged as issues of great interest to the Language Engineering community that deals with Question Answering systems. Answering questions from any text source involves the following processes:
A. Mapping the text question into a logical search query
B. Mapping the source text into a set of logical structures which the query then searches.
C. Generating the answer in text.

In this work, we dissociate the question- and answer handling, which is in the language of the query, from the logical structure generation, which requires a more comprehensive handling of the source language. In order to test this hypothesis, we have built a system that works on documents relating to a small domain -- "water and health" – and demonstrates multilingual Question-Answering, with current implementations for source text in English and questions posed in Hindi and English. Unlike traditional approaches involving translation of queries into the target language or the document collection into the source language, we convert queries and the documents into an intermediate representation, an inter–lingua called "the Universal Networking Language" (UNL) [2]. Our choice of the inter-lingua depended on the functionality and the language independent features it provided. With its set of "Universal Words" having well defined universal interpretations and ontological information embedded in them, a small and simple binary predicate structure and a knowledge base connecting the UW's as a weighted graph of relations, UNL proved to be the most fit. A general-purpose UNL enconverter and deconverter is being built for Hindi [6] and can in the future be used for expanding this system to general queries.

2. System Overview

Figure 1 on the next page gives a schematic overview of the system Two major tasks are involved in the system implementation. The first is that of ***document processing*****,** which is to be done once for each document we wish to be queried. This converts the document into the interlingua, with added inference rules. The other involves ***query processing and answer extraction*** and is done once for every query. Our present work is based on an English corpus for safe drinking water built from documents obtained from the official websites of EPA and WHO [1].

3. Document Processing

The corpus is manually annotated and processed through a Predicate Generator (the UNL Parser) to get the complete set of semantic predicates for the document (the "UNL expression" for the corpus). Some common known facts are also added to provide context information which would be commonly knownto the human reader but is not available from the text itself.

An inference engine then augments it with inferences obtained by applying a set of First Order Logic rules(inferences) to sets of predicates. For

example, given the facts Water- borne diseases are caused by ingestion of contaminated water. and cholera is a water-borne disease., one may infer that cholera is caused by ingestion of contaminated water. A meta-rule for this situation, incorporated as part of the inference rule base is that, given:
agt(cause(icl>abstract thing).@entry:1);
obj(cause(icl>abstract thing).@entry:2);nam(2:3);
which says that variable 1 causes 2, and 3 is a type of 2. Given this set of UNL relations, one can infer:
agt(cause(icl>abstract thing).@entry:,1:);
obj(cause(icl>abstract thing).@entry:,3:);
i.e. 3 is caused by 1.

4. The Question – Answer Module

Every query is processed to get an answer template that represents the form of the potential answer corresponding to the question. This template is input to an HPSG Parser [3] which outputs a pseudo UNL expression corresponding to the proposed answer template. The pseudo UNL expression is subsequently subject to a structure matching with the UNL document. The multilinguality of the system comes from the multilingual characteristic of the parser and the language – independent semantic structure provided by the inter-lingua (UNL).

4.1. Question Analysis and Processing – Generating the Answer Template

The goal of the question analysis phase is to determine the focus of the question and the expected type of answer for it. We use a set of transformational rules to generate an answer template that represents the form of the answer corresponding to a question, with a label "X" introduced at the "answer slot" (the position where the answer key-word or key-phrase needs to fit on). The rules are like 1:कहाँ:V(pre) > 1:X:गें:V(pre). They check for different features of the question like the presence of a particular word or phrase occurrence, words of a given part of speech or those of a particular semantic category. They are mostly language dependent. We have used a rule base of around 50 transformation rules for English and around 20 rules for Hindi, which has a relatively easier query structure.

4.2. From the NL Answer Template to the NL semantic predicate

The answer template is converted into a pseudo UNL representation by a multilingual HPSG Parser [3] which operates on lexicons specifying the semantic selection properties of phrasal heads. The entries in the lexicon are of the form
< बनाता >@V(sg,male){agt|~plc|~obj|~gol|!}
They capture the semantic features like agency, place, object etc. which define the UNL predicate relations. We have built separate semantic lexicons for English and Hindi with shallow syntactic rules for our restricted domain

4.3. From NL semantic predicates to the inter-lingua

The set of predicates obtained thus far have the attributes in the Query Language. In this phase these NL words are mapped (using the UW Dictionaries) to Universal Words to produce the UNL semantic predicates, very similar to the form in which the document to be queried had been converted using manual annotation and predicate generation So it is in this phase that the query (or the Answer template) representations, which had so far been in different languages, are brought to a homogeneous language-independent encoding, suitable for search. There exists a UW dictionary for every language with entries like-
[बनाता] "make(agt>thing, obj>thing)" (SG,MALE) <HINDI>
The mappings from NL words to UWs leads us to UNL predicates for the answer templates which prove to be same for similar queries in different languages.

4.4. The Search Engine

This set of predicates is now matched in the UNL Document of the corpus (the inter-lingua representation of the corpus) to give us the entities which are contenders for fitting in place of label X in the Answer Template. The search cuts down to finding a sub-graph in a graph built for the document, by representing the arguments of a binary predicate as two nodes, connected by a link with the label of the predicate, which is a UNL relation.

4.5. Conversion of the Answer from the inter-lingua back to NL (De-conversion process)

If the answer returned is a word (as a replacement for X in the UNL Document), then the de-conversion from the Universal Word to the NL word can be done directly by finding the reverse mapping from the Dictionary. In case of phrases as answers, we either have some phrasal translation directly in the UW Dictionary or have shallow de-conversion rules like mod(X,Y)=Y|X which means that a modifier relationship between X and Y would be "the word Y followed by the word X" in natural language.

The answer obtained from this phase can directly be inserted in place of the label X in the answer template and reported. In case of multiple answers matching a given set of predicates, all the results are reported.

5. An Example

Here is a sample question and its answer, along with the various stages in processing.

Source Text
"At some level, minerals are considered contaminants that can make water unsafe"

Annotated Text

<c>At some{<qua,>n} level.n.@pl.@entry </c>
{<man ,>p}mineral.@pl{<gol,>p} are consider.p
contaminant{1}.@pl{<obj,<p}that{<1}{<agt,>p}
can make.p.@possible water{<obj,<p} <c>
unpalatable{<or,>p} or even{<man,>p} unsafe.p.@
entry</c>{<gol,<p}.

Source Text in Inter – lingua

obj(consider(icl>think(agt>volitional thing, gol>uw,
obj>thing)), contaminant.@pl)
man(consider(icl>think(agt>volitional thing,gol>uw,
obj>thing)), :01)
gol(consider(icl>think(agt>volitional thing,gol>uw,
obj>thing)), mineral(icl>matter).@pl)
qua:01(level.@entry.@pl, some(mod<thing):)
gol(make(agt>thing,obj>thing).@possible,:02)
obj(make(agt>thing,obj>thing).@possible,water(icl>liquid
)
agt(make(agt>thing,obj>thing).@possible,contaminant)
or:02(unsafe(aoj>thing).@entry,unpalatable(aoj>thing))
man:02(unsafe(aoj>thing).@entry,even)

The Question(s)
जल को कौन असुरक्षित बनाता है ? (What makes water unsafe?)

The Answer Template(s)
जल को X असुरक्षित बनाता है। (X makes water unsafe.)
generated from the transformational rule(s)
1: कौन:2 > 1:X:2 (and what:V:1 > X:V:1)

The Parsed Output
(((*जल _agt(sg){}* (* को _prep{}*)) (
X_obj(){})(*असुरक्षित_gol{}*)*बनाता
V(pre,sg,male) {}*)*है _aux{}*)
+- जल कोX असुरक्षित बनाता है
+- जल कोX असुरक्षित बनाता
|+-जल को
||+-जल _agt(sg)
||+- को _prep
|+-X_obj(sg)
|+- असुरक्षित _gol
|+- बनाता _ V(pre,sg,male)
+- है _aux

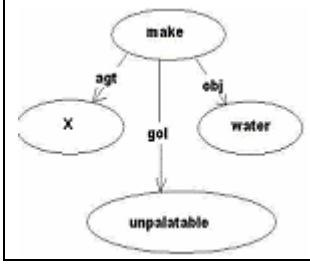
The NL Semantic Predicates

agt(makes,X)	agt(*बनाता*,X)
obj(makes,water)	obj(*बनाता*, *जल*)
gol(makes,unsafe)	gol(*बनाता*, *असुरक्षित*)

The UNL expression for the query (for both English and Hindi)

agt(make(agt>thing, obj>thing). @present, X)
obj(make(agt>thing, obj>thing). @present ,
water(icl>liquid))
gol(make(agt>thing, obj>thing,), unsafe(aoj>thing))
obtained from UW dictionary entries like –
[जल] "water(icl>liquid)" (NOUN,SG) <HINDI>
[असुरक्षित] "unpalatable(aoj>thing)" (ADJ) <HINDI>

The UNL Graph for the query



The search in the graph yields the node with "*contaminant (icl> matter)*" as a match for "X".

The De-conversion back to NL

The reverse mappings from dictionary entries like – [प्रदूषक] "contaminant(icl>matter)" (N) <HINDI> allow the answer to be converted back to the query language.

The Answer(s)

जल को प्रदूषक असुरक्षित बनाता है (Contaminant makes water unsafe)

6. Conclusion

Being our first attempt at developing a Question Answer module, the focus was more on system implementation than on system tuning. The performance of the system, for any language, will critically depend on the effectiveness of the en-conversion (NL to interlingua) and the de-conversion (interlingua to NL) schemes developed for that language. Our present module gives highly accurate results to the questions which have answers directly taken from the corpus, both for Hindi and English. This accuracy decreases with the complexity of the question and the answer, particularly due to insufficient conversion rules.

10. References

[1] Documents on Water Sanitation and Hygiene "http://www.who.int/inf-fs/en/fact112.html", "http://www.epa.gov/safewater/dwh/contams.html".

[2] Uchida,H.: "UNL Beyond Machine Translation." *International Symposium on Language in Cyberspace,Seoul,Korea(Sept 2001).(*www.undl.org)

[3] D.,Vikram,K.,Mittal,M.R.,Mukerjee,A.,Raina, A.M. "Saarthaka-A generalized HPSG parser for English and Hindi", *Recent Advances in Natural Language Processing-Proceedings ICON-2002, Mumbai, India, 18-21 Dec,2002.*

[4] Moldovan,D.,Pasca,M.,Harabagiu,S.,and Surdeanu, M. :"Performance Issues and Error Analysis in an Open-domain Question-Answering System." *40th Annual Meeting of the Association for Computational Linguistics(ACL), Philadelphia, July, 2002,* 33- 40.

[5] Martins,R "UNL as a linguistic point of view".*ICUKL 2002, Goa, India (25-29 Nov,2002).*

[6] [PPT] www.cfilt.iitb.ac.in/westernmeetjune12/ hindianalysis.ppt

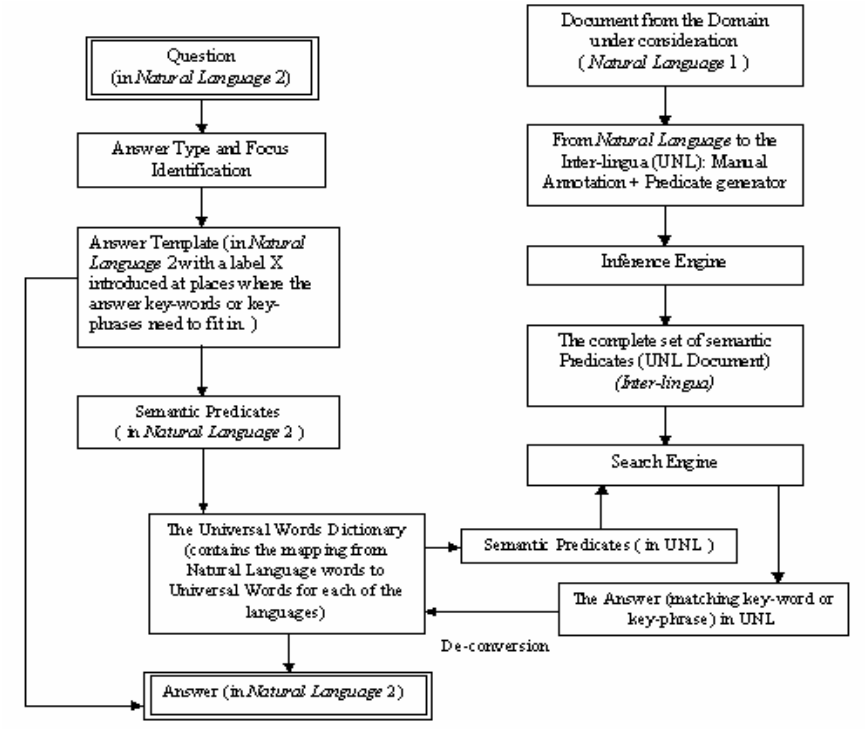


Figure 1. The System Implementation Flowchart

Predictive Texting for Tamil

A G Ramakrishnan, K E Pradheep[#] and Vaishnavi Ramaswamy
Department of Electrical Engineering, Indian Institute of Science
[#]Department of Computer Science, BITS, Pilani

Introduction

The goal of the project is to design all the aspects necessary for a viable, easy and efficient input method for SMS in Tamil language on mobile phone keyboards. This involves a judicial mapping of the Tamil alphabets to the number keys, creating a table of possible words and their frequencies, a GUI to test the system on the PC and to develop or use other necessary tools. Since the field is relatively unexplored, particularly for the language domain in consideration, we have developed most of the other tools also as the existing tools are not extensible.

Keyboard Mapping

Mobile Phone Keyboards have a limited number of keys (roughly 10). However, the use of the mobile phone as a messaging instrument (using SMS) has risen in the recent past. Currently, predictive texting schemes exist for English and many European and other languages. However, such a scheme does not exist for most Indian languages, including Tamil. Moreover, the mobile phone keyboard for the English Language is generally arranged in the alphabetical order, that is the key for the number 2 contains the letters A,B,C; that for number 3 contains D, E, F; and so on. Following a similar arrangement is a relatively inefficient scheme for Tamil; hence, we have come out with a more efficient arrangement that does not club frequently used letters on the same key.

Our mapping assigns, in general, one vowel and three consonants to each key. Consonants that co-occur have been assigned to the same keys, which we think will reduce the effort in keying in words. A word lexicon that contains the words followed by their frequency tag (its frequency in the corpus used), is used at the back-end for the prediction. A morphological analyzer has also been designed for Tamil. Based on considerable experimentation, and investigation of the results, we shall decide as to whether a morphological analyzer needs to be incorporated to provide us with better results. That is, instead of tagging just the word, we could tag its root word as well.

GUI Design

For the purposes of experimentation, we have developed a GUI that simulates the mobile phone keyboard. The GUI is entirely programmed in Visual Basic. The software provides an area for input, where the users can enter numbers and use arrow keys to move forward and backward. Alternately, the users can use the mouse for input. The GUI supports two modes, as available in most mobile phones: the smart mode, and the “a-i-u“ mode. The smart mode is the default mode. In this mode, the user needs to press the key only once, and the predictive texting algorithm will guess the most probable word and the next choices. The “a-i-u“ mode is the alternative mode, where the user may have to press a key a specific number of times to obtain a given alphabet. In both the modes, the user does not have to press the right arrow key to move to the next character. It happens automatically, if he/she waits for a few (currently 5) seconds (programmable). To handle both the modes, we have designed a Finite Automaton that converts the raw input (the sequence of user key-strokes) to a more processable form.

Prediction Algorithm

We have used the Emille Monolingual Tamil Corpus for the frequency tagging. The Emille corpus is encoded in Unicode font, and to read this, we first convert it to Romanized Tamil. Romanized Tamil is convenient and acts as a good middle medium between the Tamil Script and Unicode. For Romanized Tamil, we are currently using the Wx notation. The encoding scheme is given in a separate input file. Thus, the code can act as a generic font converter too, if provided with the encoding scheme. The corpus is tagged and pairs of words (in Romanized Tamil) and their frequencies are stored in a separate file. Internally all the processing of

Tamil words happens in Romananized Tamil, and only at the stage of the final output, the conversion to Unicode characters happens. The dictionary has been created to reside at the back-end, which simply contains each word, followed by its frequency of occurrence in the corpus. Our algorithm for predictive texting first generates all the possible combinations from the input, then compares them with all substrings in the dictionary of the same length, and screens them based on a user-defined frequency cut-off.

There is an issue to be considered here. Any Tamil input is converted so as to conform to the ISCII standards. Three important rules are: any consonant followed by a vowel is combined with the consonant itself, that is, ka + u will become ku; if any consonant that is repeated twice consecutively, a diacritic is added automatically to the first one, that is, ka+ka becomes kka; and nasals followed by consonants get combined, that is gna + ka become gnka. These rules prevent us from doing a direct search on a dictionary as a case might arise when the user enters the digit corresponding to 'ka' twice. We cannot simply continue with a set of words that end with just 'ka', because we might lose words which end with 'k'. Thus, we have to use a look-ahead to solve this problem. However, in this case, it can be observed that it is sufficient to retain just the root of the consonant; that is, instead of ka, we retain "k", and then screen out words which do not end with "k". The set of ISCII rules is easily configurable, as they are stored in a separate file.

A file contains the number-to-alphabet mapping, which can be changed for experimentation purposes. We are using an incremental algorithm to generate all the possible combinations. Since frequency cut-offs will reduce the number of words at any stage, this algorithm works considerably faster than a brute force algorithm. The word combination with the highest frequency is selected to be displayed automatically. The user can change the word by selecting a different word or spelling out the word. The words are displayed in Tamil Script using a Unicode Font. The words are displayed in Romanized Tamil as well.

Results

We have now obtained the revised Emille corpus, and we assume that most of the errors in the initial corpus have been eliminated. By using the revised corpus, and by analysing the entire corpus, the performance of the system has improved. Though the system predicts common words, the performance of the system depends on the corpus used. We have picked disparate areas of the corpus, such as, politics, sports, cinema, etc. for the frequency tagging.

With this project, the other tools we developed could be used in other contexts as well, with little or no change. A generic font converter could be developed. The code already handles Romanized English to Tamil Script. This would be helpful for people who do not know Tamil, want to type in Tamil. Moreover, the Predictive Texting Scheme could be applied in Word Processors, as usage of Tamil in Word Processors is becoming increasingly abundant.

The project could easily be extended to other Indian languages, as it does not assume anything about the language. The requirements are a set of Unicode rules, a corpus (or a word-list with frequencies) and a set of ISCII rules (optional). The number-to-key mapping is easily configurable, and can be experimented with to come up with an efficient mapping.

Directions for future research are in the areas of predictive texting with syntactic analysis, and next-word-prediction with semantic analysis.

POSTERS

Analysis and Design of Oriya Morphological Analyser : Some Tests with OriNet

Sanghamitra Mohanty
sangham1@rediffmail.com

Prabhat Kumar Santi
pksanti@rediffmail.com

K.P. Das Adhikary
krushnapada@rediffmail.com

PG Department of Comp. Sc. and Application
Utkal University
Bhubaneswar, Orissa, India- 751004.

Abstract

Indian languages are characterised by a rich system of inflections (VIBHAKTI), derivation and compound formation for which a standard Morphological Analyser (MA) is needed for a machine to deal with the lexicons of those languages [2]. The numbers of word have been derived from the root word by some specific orthographic rule in the Indian languages. This paper deals with the analysis and design of Oriya Morphological Analyser (OMA). The lexical category and computational grammatical model of Oriya language are described to build a complete OMA in the current scenario. The major contents on which our OMA has been built up with i) Pronoun Morphology (PM), ii) Inflectional Morphology (IM) and iii) Derivational Morphology (DM) [7]. The OMA system is designed according to the object oriented approach to increase its reusability, robustness and extensibility. We have developed and implemented the Decision Tree (DT) and its respective algorithm of each type of morphology, through which our OMA runs successfully. Some tests of the OMA with OriNet: the Word-Net for Oriya Language has been reflected in this paper.

The present work aims to build a computational model for the analysis and generation of Oriya language, the language being one of the official languages of the state of Orissa, situated in the eastern part of India. An important advancement of computerized language generation and understanding is to capacitate Morphological Analysis (MA) for any type of lexicon. For example, Oriya words like, ବାଳକମାନେ *(bAlakamAne)* (boys), the system technically reduces it to root word ବାଳକ *(bAlaka*) (boy) and at the same time provides other information like part-of speech (i.e., noun), suffix (ମାନେ), gender (i.e., masculine), number (i.e., plural) and case-ending relation (i.e., subjectivity). This analysis is performed by our OMA, which not only deals with the study of words but also its morphemes. A morpheme is defined as the minimal meaning-bearing unit in a language. Morphemes vary from language to language and when it is added to the root word, denote different meaning. It can be divided into three types such as prefix, suffix and infix. Prefix and suffix precede root word whereas infix gets inserted inside the root word. A word can be easily identified by the system as a sequence of characters delimited by spaces, punctuation marks. A word can be of two types as simple and compound. A simple word consists of a root or stem together with suffixes or prefixes. A compound word can be broken into two or more simple words.

The OMA system is needed for various applications in developing NLP tools [1,3,5]. For example, in OMT, there is need of root words, which is obtained through the OMA. In the words like ଯାଇଛି *(jAichhi)* (has/have gone), ଯାଇଥିଲି *(jAithiLi)* (had gone) etc., OMA provides both the information of root word as ଯା *(jA)*(go) and suffix as ଇଛି (*ichhi*) and ଇଥିଲି (*uthili*) to the OMT. Similarly, it has typical use in OriNet for searching any type of lexicon.

In the process of forming words in Oriya language there exists three major classes of morpheme such as [7] Pronoun Morphology (PM), Inflectional Morphology (IM), Derivational Morphology (DM)

The PM is the study on the grammatical classification of pronoun. For example the pronoun ତୁମ୍ଭେମାନେ *(tumbhemAne)* (you) indicates that it is personal pronoun, 2nd person and plural number. The PM of Oriya language has been classified into different groups according to their syntactic rules. Some of the pronouns are ambiguous as in the interrogative pronouns like କିଏ *(kie)* (who) is used as plural as well as singular. For these ambiguities, further modification is necessary.

The IM is the combination of root word with grammatical morphemes, usually resulting in a word of the same class as the original stem and filling some syntactic function like agreement. In other words, the morpheme, which conveys ‘number’,

'person', 'inflection', '*kAraka*' etc., is called IM. For example the inflectional morphemes ମାନେ *(mAne)* for making plural form (number), third person (person), subject (*kAraka*) and 1st inflection (inflection) etc., in case of nominal word. Similarly, from the inflectional morphemes like ଉଥାନ୍ତି *(uthAnti)*, we obtain information as present tense, 1st person and singular number in case of verbal word. Like Sanskrit, Oriya language has also strong inflectional system, which can be classified into two classes such as nominal morphemes and verbal morphemes. There are nearly 40 morphemes for nominal words and 100 morphemes for verbal words in Oriya. The nominal morphemes are attached to nominal words, whereas verbal morphemes are attached to verbal words. All the suffixes in IM determine the number, person, case-ending relation, tense and inflection of the word. The verbal morphemes are quite complicated than nominal, which are classified according to the tense, person, and number attached to the verbal word only. Moreover, there are also some ambiguous morphemes in both the cases of nominal and verbal as in the nominal morpheme ଙ୍କୁ *(Ngku)*, which in one hand indicates 2nd inflectional singular number on the other hand 4th inflectional singular and plural number. But, our Morphological Parser (MP) successfully handles them and provides different alternatives too.

The DM is the combination of a word stem (root word) with a grammatical morpheme, usually resulting in a word of different class, often with a meaning hard to predict exactly. The prime characteristic of this morphology is that even if it doesn't imply any number, person etc., like IM, but has an important role to convert from one lexical category to other. For example a verb word ହସ୍ *(Hasx)* (laugh) can take the derivational suffix i.e., ଅଇବା (*eibA*) to produce a nominal word as ହସେଇବା *(HaseibA)* (making laugh). Similarly, also a noun word (*ANThu*) is converted to a verb word as (*ANTheibA*) by use of DM i.e., ଅଇବା *(eibA)*. The DM is quite complex but it is handled easily by OMA. There are so many words belonging to verb group derived from nominal words and also words belonging to verb group derived from verbal words. Generally, we find two types of DM such as prefix and numberless morphology in Oriya. The prefix can be further classified into two groups such as modified prefix (modifies the meaning of the root word) and negative prefix (changes to the antonym of the root word). Similarly, the numberless morphemes can be classified into five groups such as VsN (Verbal Noun), VsV (Verbal Verb), Gs (Gender suffix), NsV (Nominal Verb) and NsN (Nominal Noun) [4,7]. For examples, the word ପରାଜୟ (*parAjaya*) (defeat) is derived from the negative prefix ପରା (*parA*) and the word ଜୟ (*jaya*) (win).

The architecture of OMA is divided into five parts (Figure-1) such as OriNet Database (OD), which stores the Oriya lexicon (Only root words) whereas OMA Engine (OE) processes the system and Morphological Parsing (MP) parses the word according to orthographic rule. Decision Tree (DT) decides to classify the morphemes and different functional programmes by use of OMA[6].

The OMA system has been designed on the basis of Object-Oriented Approach (OOA). By use of this design methodology, different functions can be added or deleted to the existing system conveniently. We have implemented Pronoun Morphology and Inflection Morphology in the OMA in such a manner that it is successfully run with the OriNet system (Figure-2), Oriya Spell Checker (OSC) and Oriya Grammar Checker (OGC) [1,3,5]. The OSC handles any type of word (derived, inflectional or root) using the OMA

It also provides sufficient interface for applications involved in Oriya Machine Translation (OMT), Word-Net for Oriya (OriNet), Oriya Spell Checker (OSC) and Oriya Grammar Checker (OGC) [4,5]. All these developments have been worked out on the basis of the syntactic approach of Sanskrit language for which, we hope the technology involved here can be extended to any other Indian languages.

Keywords: Word-Net, OriNet, Morpheme, Inflectional Morphology, Derivational Morphology and Pronoun Morphology

Reference

[1] Alam. Y. S. et. al. " *Lexicons in an Object-Oriented Grammatical Model For Universal Grammar – Based Machine Translation (UGMT)"*. Proceedings of the 1st GWN conference January 21-25, 2002, CIIL, Mysore.

[2] Bharati. A. et. al. (1995) ."*Natural Language Processing, A Paninian Perspective"*. Prentice Hall of India Private Limited, New Delhi-110001.

[3] Chadhuri B. B. et. al. "*To wards Indian Language Spell Checker Design"* IEEE Proceedings of LEC-2002, University of Hyderabad, Hyderabad, India.

[4] Fellbaum C. Editor (1999). "*WordNet: An Electronic Lexical database"*. The MIT Press, Cambridge, Massachusetts, London, England.

[5] Mohanty S et al " *Object Oriented Design Approach to OriNet System: On-line Lexical Database for Oriya Language"*. IEEE Proceedings of

LEC-2002, University of Hyderabad, Hyderabad, India.
[6] Rayner D' Souza et al. " Natural Language Generation from Semantic Net like Structures with application to Hindi", STRANS- 2001, IIT Kanpur, Kanpur, India.
[7] Mohapatra Pandit N. and Dash S.(2000). "*Sarbasara Byakarana*", New Students Store, Cuttack, Orissa, India

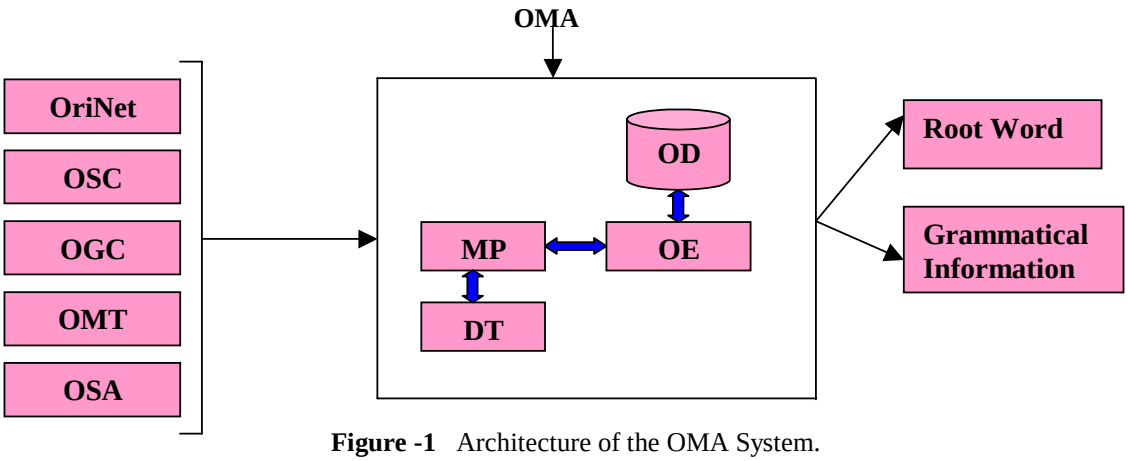


Figure -1 Architecture of the OMA System.

(OD = OriNet Database, OE = OMA Engine, MP = Morphological Parsing, DT = Decision Tree, OriNet = WordNet for Oriya, OSC = Oriya Spell Checker, OGC = Oriya Grammar Checker, OMT = Oriya Machine Translation and OSA = Oriya Semantic Analysis)

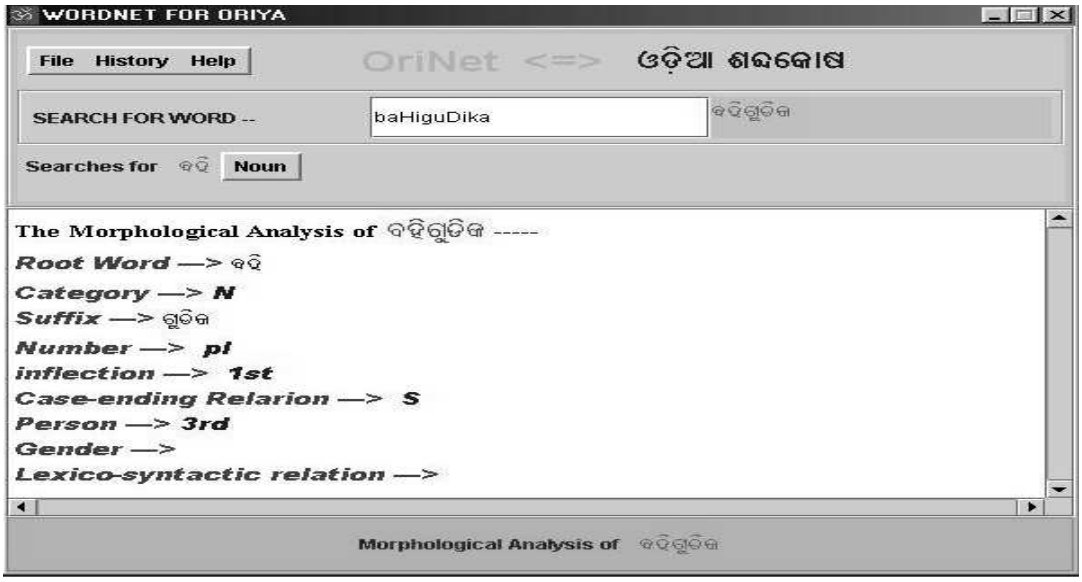


Figure 2:- Output of the OMA of the word ବହିଗୁଡିକ *"baHiguDika"* (books) in OriNet. (N = Noun, Pl = Plural, 1st = 1st inflection, S = Subject, 3rd = 3rd person)

Ideas to Develop a Morphological Parser for Bangla

Joy Mustafi
Department of Computer Application
RCC Institute of Information Technology
Kolkata – 700015. India
must143@rediffmail.com

1. Introduction

In all Modern languages, where there is a rich cultural heritage of multiple millenia, it is a problem to find a well-defined system of derivational morphology. In Bangla, the problem in this respect is no less than in any other language. The problems with Bengali derived words are the following.

Firstly, Sanskrit words were not borrowed systematically. Sometimes, we find that a Sanskrit root is absent but a derived word is present. Hence we have the word *paryataka* (a traveller) but not the root *at* (to roam), and so there's no way to know that it is a derived word in a synchronic analysis of the language. Again, we find the root word is present but the derived word is absent. Thus we see that, the roots are present, and the suffix 'ana' denoting the instrument is absent for most of the roots but present only in one or two cases, and means something other than the instrument.

Secondly, some words in Sanskrit had a connection with each other, but the forms of these words have changed over centuries and semantic shifts have occurred and in Bengali, there's no way to see that they are connected. Thus, the words, *karoti* (does) and *karta* (the do-er or the agent) in Sanskrit clearly have a relation, but in Bengali there's no relation between the words *katta* (derived from Skt. *karta*, meaning the husband or the lord), and the word *kare* (does).

Thirdly, foreign words complicate the scenario:

i) foreign prefix/suffix are added to native roots and vice versa
ii) foreign derived words are borrowed without foreign roots.

However, like most other Indian languages, inflectional morphology is extremely well developed. We would therefore deal only with inflectional morphology.

2. Bengali Morphology

The nominal stems were inflected in Sanskrit into different cases and numbers. In all, there were thirty suffixes, of which at least five were repetitions. In Bengali, two of the cases have merged, and the use of auxiliary words has brought down the number of distinct suffixes to five or six.
The tenses and moods and the different persons for each of these make the number for inflection of verbs to about fifty. A suffix for tense or mood is attached first and then a suffix for person.

3. The Algorithm

The tense/mood information and the person information is extracted for the verbs together with the verb root denoting the principal action, and for the nouns, we extract the nominal stem, the number and the case information. Our algorithm would have methods to deal with the alternative forms of verb roots in different cases without much difficulty, and having a small size for the dictionary at the same time.

a) Verbs

- Extract the suffix for person by matching.
- Extract the suffix for tense or mood (or case) by matching.
- Search for the remaining in the root directory,
- If error at any stage, go back and try other options

b) Nouns
- Extract the suffix for case/number by matching.
- The remaining portion is the nominal stem

Example
Noun: *chele/ke*
Verb: *dekh/l/am*

4. Limitations

Since we propose to deal with written language or spoken language after conversion to text, hence we would consider orthographic spellings in our paper. We would assume that the sentences are in textual form and are spelled correctly.

5. Advantages

The number of verb roots in Bangla is in the order of several thousands. So we save ourselves from keeping about a hundred thousand verb forms in the dictionary. The nouns are numerous. So, the savings there is just huge.

References

[1] Aronoff, M., "Word Formation in Generative Grammar", Mass: MIT, Cambridge, 1976.

[2] Austin, J. L, "How to Do Things with Words", Oxford: Clarendon Press.1962.

[3] Miller, G. A., "Wordnet: A Dictionary Browser in Information in Data",
Proceedings of the First Conference of the UW Centre for the New Oxford Dictionary. Waterloo, Canada: University of Waterloo, 1985.

[4] Miller, G. A, "Dictionaries in the Mind", Language and Cognitive Processes 1:
1986, pp. 171-185.

[5] Beckwith, R., Fellbaum, C., Gross, D., and Miller, G. A. "WordNet: A Lexical Database Organized on Psycholinguistic Principles", U. (ed.).

[6] Erlbaum, N.J., "Using On-line Resources to Build a Lexicon". Hillsdale.

A Frame-Semantic Approach for Tagging Hindi and Bangla Sentences

Pankaj Goyal, Ankit Soni, Deepak Sharma,
Amitabha Mukerjee, and Achla M Raina
Indian Institute of Technology, Kanpur,
Kanpur 208016,
{pankajgo, ankit, deepaks, amit, achla}@iitk.ac.in

Extended Abstract

This work proposes an approach to Semantic Tagging for Hindi and Bangla based on Frame-Semantics (Fillmore, 1982 and others). We take up a set of language-independent conceptual Frames, such as the frame INGESTION, with core frame elements (FEs) Ingestibles, and Ingestor.

A frame is an intuitive construct that allows us to formalize the links between semantics and syntax in the results of lexical analysis. Semantic frames are schematic representations of situations involving various participants, props, and other conceptual roles, each of which is a **frame element (FE)**. The semantic arguments of a predicating word correspond to the frame elements of the frame (or frames) associated with that word.

We associate frames with lexical units from Bangla and Hindi -- Table 1 lists some lexical units for the frame Ingestion along with their English equivalents:

Table-1

English	Hindi	Bangla
breakfast.v	नाश्ता	prAtarAsh v
Consume.v	भोग करना	bhog k.v
drink.v	पी	khA.v
eat.v	खा	khA.v
feast.v	भोज करना	bhoj k .v
feed.v	खिला	khAoyA.v
gulp.v	निगल	gelA.v
have.v	ले	Neo.v
munch.v	चबा	chebA.v
nibble.v	कुतर	ThokrA.v
sip.n	घूँट	chumuk.n
sip.v	घूँट लेना	Chumuk de.v

The semantic combinatorial possibilities of the lexical items are documented through manual semantic annotations of sentences extracted from a large corpus.

In addition to the core FEs (which do not appear in any other frames), lexical units in a frame may also admit peripheral and extra-thematic FEs. For example, the word ``eat'' in the frame Ingestion, can take the peripheral FEs Instrument, Manner, Place, Source, Time, etc.

An annotated example from this frame follows:

As the house doesn't have a dining room, *the family* [EATER] *eat* [lexical unit] *in the large kitchen* [PLACE].

क्योंकि घर में भोजन कक्ष नहीं है , *परिवार के लोग* [EATER] बडे *रसोयी* [PLACE] मे *खाते हैं* [lexical unit] ।

bARite bhojan kakSha nei tAi paribArer sabAi [EATER] rAnnAghare [PLACE] khAy [lexical unit].

The frame-semantic construct of frame inheritance including multiple in heritance is applied to lexical units drawn from the two languages. The larger aim of the project is to build lexical databases for Indian Languages, which make use of language-independent semantic frames for tasks such as multilingual natural language understanding, translation, dialogue systems, ellipsis resolution, etc.

Reference:

1. Baker, Collin F. and Hiroaki Sato . *The FrameNet Data and Software*, Poster and Demonstration at Association for Computational Linguistics, 2003, Sapporo, Japan

2. Lowe, J.B., Baker, C.F. and Fillmore, C.J. (1997): A frame-semantic approach to semantic annotation, in *Proceedings of the SIGLEX workshop "Tagging Text with Lexical Semantics: Why, What, and How?"* ,April 4-5, Washington, D.C., USA in

Parsing a Free Word Order Language: Tamil

B Kumara Shanmugam
AU-KBC Research Centre
MIT, Anna University,Chennai, India
bkumar@au-kbc.org

1. Introduction

Parsing is the task of assigning an appropriate syntactic structure to a sentence of a language. Tamil is a relatively free word order language and it belongs to the Dravidian language family. Parsing free word order languages is still an open problem and many researchers have proposed different approaches for this issue. The system that we present here is a rule-based system that uses our grammar formalism and a few other components to analyze sentences. The input to the parser system is a Tamil sentence and the output is the parse structure of the sentence.

2. Morphological Analysis

The input Tamil sentence has to undergo different levels of preprocessing before it could be actually parsed by the core parser. At the first phase, the given sentence will undergo morphological analysis to identify the root word and the suffixes of each word of the sentence. Morphology of Tamil is agglutinative and it is realized through suffixation. The morpho-phonemic (sandhi) changes that take place during suffixation complicate the analysis process. The suffixes are representation of the grammatical features of the word form and the root word represents the lexical meaning of the word form. The root along with suffix helps in identification of the grammatical category of the word. The morph that we have developed has a coverage of 96% over the 3 million word Tamil corpus developed by CIIL.

Consider the word "kettAnY"(he heard) the output of the morphological analyser would be "kelY<verb>+tt<past tence>+AnY<3rd person, singular, masculine>" and for the word "marawwE"(tree+accusative_case) the output will be "maram<noun>+E<accusative case>", Where the final 'lY' of the root word 'kelY' changing to 'tt' and 'm' of 'maram' changing to 'ww' are sandhi changes. For processing purpose we represent the features as binary values, hence for the above two words the tags will be

kettAnY<v:+p,-a,+1,+2,+s,+m,-f>
where v => verb
+p,-a => present tense
+1,+2 => 3rd person
+s => singular
+m,-f => masculine gender
marawwE<n:+acc>
n => noun
acc => accusative case

3. Part of Speech Tagging

There may be ambiguity in identifying the features or the part-of-speech (grammatical category) of the word, which the morph analyser cannot resolve. To resolve such ambiguities, in the next phase we use part-of-speech Tagger. A Tagger takes the output of the morph analyser and resolves the ambiguities using lexical level rules and/or context level rules to whatever extend it can do, the rest has to be taken care by the parser. The disambiguation performance of the Tagger is 98% over the ambiguous tags generated by the morph. The morph output for the words "eVlYYuwiya" (written_adjectival participle) and "eVlYYuwa" (to write_infinitive) will be

evlYYuwiya => eVlYYuwu<v> + i<past tense> +a<infinitive | adjectival participle>
eVlYYuwa => eVlYYuwu<v> + a<infinitive | adjectival participle>

Here the suffix 'a' could be an infinitive suffix or an adjectival participle suffix. The tagger will resolve this using the lexical rule that, for a adjectival participle suffix there should be an preceding tense marker. Hence the first word is an adjectival participle and the second word is a infinitive form of verb.

4. Normalization

After identification of the right part-of-speech and features for each word in the sentence, the sentence can be parsed in a typical parsing system. In our system we are introducing a normalization phases in the stream. If we want to cover a free word order language syntax in a generic phrase structure grammar the permutation of the rules will increase the number of production rules. To avoid this, we descramble the sentences so that the phrases of the sentence are in specific order so as to keep the number to rules to minimum and to reduce the load on the parser in identifying the arguments for the verbs. The normaliser identifies the phrases in the sentences and descrambles them. The coverage of the normalizer is 72% without named entity recogniser. If we do manual or semi-automatic named entity recognition the coverage shoots up to 95%.

For the sentence

paricE manwiri kulYYanwEkku koVtuwwAr

(gift+acc) (minister) (child+dat) (give+past+3sh)

the normaliser output would be

manwiri kulYYanwEkku paricE koVtuwwAr

5. Parsing

Finally the sentence goes to the core parser for identification of the syntactic structure. The algorithm for parsing is a modified chart-parsing algorithm. The parser checks the argument for verbs and for feature unification. The core parser takes the output of the normalizer and, uses the phrase structure rules and argument information of verbs to assign appropriate syntactic structure to the sentence.

Some of the rules of the phrase structure grammar are

S => *NP *VP

NP => *N

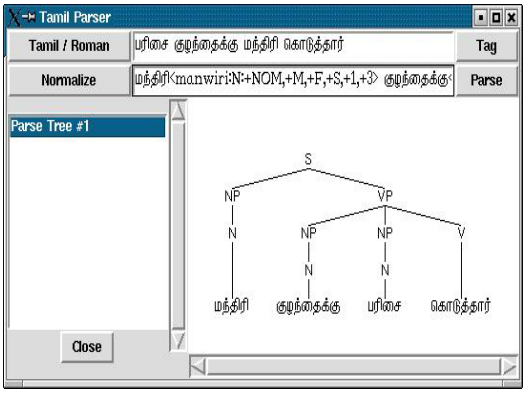
VP => NP NP *VP

The subject is always nominative case in Tamil with a few exceptions. The argument structure for the verb 'koVtu' (give) is <–dat +acc>, implying that it takes a mandatory accusative argument, preceded by an optional dative argument. The argument structure and feature unification has to be done for the symbols that are preceded by '*'.

The output sentence structure from the parser can be represented in a linear form as

S[NP[N[manwiri]] VP[PP[[kulYYanwEkku]] NP[N[paricE]] V[koVtuwwAr]]

or as a parse tree.The screenshot for the same would be



The parse representation thus generated will be directly useful for many language processing applications like Machine Translation, Information Retrieval etc. The above system has been implemented to handle sentences with embedded clauses. It is capable of handling infinite, verbal participle, conditional and relative clauses. Using the corpus argument structure for most frequently occurring 40 verbs has been identified which can be extended for more verbs.

6. Conclusion

The methodology that we follow in this system reduces the volume of the grammar rules and also simplifies the parsing process. In this paper we have covered how this system is customized for Tamil, but the whole system is a generic one and hence it can be easily tuned for other Dravidian languages too. Generation of dictionary and rules for that particular language can do this.

Some Issues in Machine Translation and Automated Document Processing Systems

K.Chandra Sekharaiah
Associate Professor in CSE
JNTU College of Engg.
Hyderabad – 500 072
shakes123@sify.com

K.Suresh Babu
Asst. Prof. in CSE
Jyothi Engg. College
Hyderabad.
gopal_u@sify.com

Upakaram Gopal
Research Associate
Pullaiah AI Research
Foundation, Nellore
gopal_uv@rediffmail.com

In this paper, we present the results of consolidation of our earlier work and updation carried out by us on schizophrenia in modeling language systems [1,2]. The nuances of understanding schizophrenia [4] in terms of the pessimistic approach in medical science versus optimistic approach in computer science are brought out. The variegated manifestations of the notion across medical, object, neuroscience and language systems are presented. Idiosyncrasies in modeling determine the characteristics of the notion. It is argued that the notion does not have an altogether negative connotation. Understanding the concept facilitates to avoid pessimistic schizophrenia in system modeling. It is believed that future work on understanding the subtleties of schizophrenia will enable the development of holistic systems.

Second, literature survey was carried out on addressing issues in the realm of automated document processing systems [3]. It is found that the present language engineering software is inadequate to process certain peculiarities in language constructs such as idioms and phrases, proverbs, etc in the documents in interlingual translation.

In spite of the challenging and potentially immense problems in the Indian scenario for successful thorough work on NLP, national effort and national will shall solve the subset of the language problems that fall within the gamut of NLP. This means that we, the Indians, should be polyglots as per the specification of the 3-language formula and resolve the problems of natural language modeling for national cause in particular and for the cause of the world in general.

REFERENCES

1. [Mandya2003] **Chandra Sekharaiah K**., Gopal U., *"Schizophrenia in System Modeling"*, AICTE-ISTE Second National Conference on Document Analysis and Recognition, (NCDAR 2003), Mandya, Karnataka, India, pp. 1—5, Mar 2003.
2. [LEC2002] **Chandra Sekharaiah K.**, and Janaki Ram D., *"The Dynamics of Language Understanding"*, in Language Engineering Conference Inter-national, (Hyderabad, India), pp.197—200, IEEE CS Press, Dec.2002.
3. [Mandya2003-2] Sekharaiah C. K., Gopal U., *"Some Issues in Document Analysis and Recognition"*, AICTE-ISTE Second National Conference on Document Analysis and Recognition, (NCDAR 2003), Mandya, Karnataka, India, pp. 6—12, Mar 2003.
4. **Chandra Sekharaiah K.**, Gopal U., K.Venkateswar Rao, "On Understanding the Social Responsibilities of Modern Information Systems Managers", ISTE National Seminar, Mar 2004 Luthiana(Accepted).
5. "The Sound System of Indian English", Monograph No. 7, Central Institute of English and Foreign Languages, Hyderabad

A Complete OCR Development System for Oriya Script

Sanghamitra Mohanty, Hemanta Kumar Behera
RC-ILTS-Oriya,
Dept. of Comp. Sc. And Appl.,
Utkal University,Bhubaneswar, Orissa,India-751004.
sangham1@rediffmail.com, kumar_hemanta@yahoo.com

Abstract

There are lots of application areas where, OCR can help. Major areas are described below.

1. Preserve old documents in electronics format.
2. Save document images within limited space.
3. Help visually impaired persons to read the content on the document.

We developed an OCR system for Oriya script. Oriya alphabets are complex in nature and consists of 115 alphabets among which around 60 characters are very difficult to recognize because they are composite characters (conjuncts). Another major problem is due to similar shaped character. For example in Oriya the letter "L" is similar with "S". A set of algorithms have been that attempts to work with a subset of features in a character that human would typically see for the identification of machine printed characters. The feature point of human interest in an image, a place where something happens i.e. it is a point which is distinguish from all other points in that image matrix. It could be an intersection between lines, or a corner, or a dot surrounded by space.

In a broad sense Document Image Processing consists of 4 steps namely

1. Preprocessing
2. Feature Extraction and Classification
3. Recognition
4. Post Processing

Preprocessing is the most important step of character recognition, which confirms the input image into a most valuable and correct format that can be fed to the Recognition Engine. This process involves several activities such as Digitization and Noise Cleaning, Skew detection and correction, Line Extraction, Word, Character Extraction and *matra* extraction., Thinning and Zooming.

An optically scanned document image is usually an integer (gray-valued) array; a process of spatial sampling and simultaneous conversion of light photons to electric signals obtain it. A high-resolution *bit map* is sufficient to capture shades of gray in the eye of the human perceiver (such images are called *half-tone*). An optically scanned document image can be converted into a bit map by a global thresholding operation: pixel values below the threshold are deemed to be black (value 1) and those above deemed white (value 0). The threshold itself can either be predetermined or calculated from the image histogram. e.g the valley of a bimodal histogram.

The process of removing unwanted pixels from the input image is known as noise cleaning[3]. Technically, we can say noise is one of the pixel value i.e. intensity among intensity values but it has unique characteristic with respect to its neighboring pixel values i.e. intensity values. Noise arises due to so many reasons namely old document, low paper quality, and dust particles on the surface of scanner, low quality ink and low quality printing machines. Here, we developed an algorithm, which removes isolated points using global thresholding applied on the entire range of pixels and using adaptive technique.

During scanning the document may be slanted leading to problem at the time of line extraction. The first and foremost step after noise cleaning is skew detection and correction. After noise cleaning the image matrix consists of two gray values one value denotes background and other denotes foreground. In this paper we assumes 0 for foreground and 1 for background. Angular projection profiles are prepared for –8 degree to +8 degree with an interval of 0.05 degree and a strip-wise histogram is constructed using these projection profiles. Then a global maxima was found out at a particular angular direction. This angle gives the detected global skew angle and then the whole document is then rotated to get the skewless image. Line extraction is one of the segmentation technique in which individual lines are extracted for the whole document image. The whole document is a matrix of 0's and 1's where 0 stands for black

i.e. foreground pixel and 1 stands for white i.e. background pixel in which each row is a combination of 0's and 1's and the row which has less number of background pixels can be represented as the partition point. Here in our paper, we have developed the technique based on horizontal projection profile that is able to extract individual lines from any document. The histograms for each line are determined and from these histograms line borders are made out as the extreme maximas. This technique can be applied to any script but for Oriya script, we have given some constraints because Oriya alphabets are generally complex in nature. The complexity arises due to so many reasons namely the occurrence of vowels after consonants etc. and is termed as modifiers (*matra*) and has different shapes. This is not only a problem of Oriya script but also for some other Indian Scripts. Hence this type of constraint can be applied to Indian script. Modifiers in Oriya script can be appeared in four places namely left, right, upper and lower. At the time of line extraction only lower and upper *matra* or lower *phala can pose problems* and a region analysis method developed basing on thresholding to solve this type of problem. A line is a combination of words and each word extracted from the respective line depending upon the threshold on the spacing between words and each word is combination of characters and *matras* and characters are extracted by forward and backward chaining method. A major problem lies when the individual characters are connected and to solve this type of problem, the baseline characters are extracted from the line by region analysis and labeling and then a average mask is calculated based upon the individual characters. The mask is allowed to move from both forward and backward directions. Thinning is the process of extracting border pixels from the image matrix preserving its connectedness. Here the character extracted from the line gets thinned by applying the algorithm developed by Zhang and Suen (1984)[1]. A feature is a point in a pixel matrix, which is distinguished from all other pixels by its unique characteristics. The most common type of features is Structural features and Topological features. Here we used structural features for character recognition. There are ten structural features extracted from the 16x16 pixel matrix which are – Upper Part Circular, A vertical line on the right most part, Holes, Horizontal Run code, Vertical Run code, Number of holes, Position of hole.

After features are successfully extracted from the respective 16x16 pixel matrix, the process goes for classifying the extracted features. We developed a tree-based classification of method in which each node denotes a particular feature and all leaf nodes contain individual characters, modifiers (*matras),* digits and composite characters (*phalas)*. A character in the form of 16x16 pixel matrix after preprocessing is allowed to sink into the above tree and finally reach in one of the leaf node denotes a particular character. The recognition phase has two parts. In the first phase individual characters, left and right modifiers (*matra*) are recognized based on structural features whereas in the second stage upper and lower modifiers (*matra*) are recognized based on run length code. Most composite characters[1] (*Yuktas*) are recognized by applying run length code, loop and position of hole. The above classification tree, which has been developed, may be binary or tertiary. Each level containing feature is assigned to a string of 0 and 1 i.e. if a character contains a line then 1 is assigned otherwise 0, in this way 0 and 1 values are assigned to the extracted feature and finally the matrix of pixel is matched in all levels at run time and sinks to a particular leaf node which contains a ASCII and ISCII value. In each level a condition has been checked and the recognized character sinks according to the condition and then an ISSCII or ASSCII value is dumped into the editor.

Due to nonavailability of 100% noise freed digitized image and presence of similar shaped characters the accuracy rate is affected. To tackle this type of problem the system has been integrated to spell checker with the help of dictionary and a huge corpus.

References

[1] Grain U and Choudhuri B B 1998 compound character recognition by run number based metric distance. *Proc. SPIE Annual Symposium on Electronic Imaging, San Jose, USA, pp 90-97.*

[2] Digital Image Processing, by Rafael C. Gonzalez, Richard E. Woods.

[3] S.Mohanty, K.Sahoo and H. K. Behera," *A New Algorithm for the restoration of characters in old noisy document with varying level of intensities*", ISC Conference, India, Jan'2003.

[4] S. Kahan, T. Pavlidis and H. S. Bairb, "On the Recognition of Printed characters of any font and size", IEEE Trans. Pattern Analysis Machine Intelligence, Vol-9, 1987, pp. 274 -287

Calculation of MFCC from Mel-Spectrum for Speech Recognition

Sanghamitra Mohanty and Navoch Mohanayak*

Department of Computer Science and Application,
Utkal University, Bhubaneswar – 751004
*Xavier Institute of Management,
Bhubaneswar - 751013

Abstract

Speech signal has the fundamental property that it varies slowly with time, so it can be represented as a linear time varying function. The speech signal is segmented to short units and then modeled as a result of a linear time variant system by a quasi-periodic impulse or random noise signal. The excitation and impulse response of a linear time-invariant system can be looked as a case of separating the components of a convolution, so can be named as deconvolution [1]. We shall be using the Fast Fourier Transform to convert each frame of N samples from the time domain frequency domain. During spectral analysis it is observed that .Mel-Cepstrum is the inverse Fourier Transform of the logarithm of the power spectrum signal. Mel-Cepstrum can be calculated from Mel-Spectrum as an acoustics parameter during speech recognition. Mel-Cpestrum or Mel-Cepstral Coefficient can be calculated by different methods. One may be from the Linear Prediction Coefficient[2]. It can also be calculated by converting the log Mel-Spectrum back to time domain using the Discrete Cosine Transform (DCT). This will return the Mel Frequency Cepstrum Coefficients (MFCC). Applying the natural logarithm of the Mel-spectrum to get the Mel-Cepstrum, which are real numbers.

The cepstrum c[n] of a real sequence s[n] can be defined as[2, 3]

$$c[n] = (/2\pi) \int_{-\pi}^{\pi} \log | X(e^{jw})| e^{jwn} dw \ , \quad -\infty < n < \infty \qquad (1)$$

We can also calculate the approximation to the cepstrum by computing the inverse Discrete Fourier Transform of the logarithm of the magnitude of the Discrete Fourier Transform of the input sequence as

$$c_p[n] = (1/N) \sum_{k=0}^{N-1} \log | X_p(k)| e^{j 2\pi/N \ kn} \ , \quad 0 \le n \le N-1 \qquad (2)$$

To this function we apply Discrete Cosine Transform and thus get the Mel Frequency Cepstrum Coefficient (MFCC). We have experimented this with five speakers (Three male and Two female) and have tested it for the vowel “I”. The result are depicted in Table-I below. It is observed form the experimented results that the coefficient match perfectly well.

Calculating the Mel-Cepstral coefficient by this method the truncation error in the frequency domain can be minimised. This helps for the recognition of the segment of the speech signal in discrete mode more efficient.

Reference:

[1] L.R. Rabiner and R.W. Schafer, “Digital Processing of Speech Signals”, Prentice Hall, New Jersey.

[2] A. V. Oppenhiem and R. W. Schafer, “Discrete-Time Signal Processing”, Englewood Cliffs, N.J. Prentice Hall, 1989.

[3] J.W. Picone, “Signal modeling technique in speech recognition,” Proceedings of IEEE, Vol. 81, No. 9, 1993.

[4] T. Tokuda, T. Kobayashi anf T. Fukuda, S. Imai, “ An adaptive algorithm for mel-cepstral analysis of speech”, Proceedings of ACASSP, 1992.

VOWEL 'I'

Sumit		Suman		Varun		Urmila		Manmaya	
-2.3278	0	-2.7322	0	-2.1193	0	-2.4702	0	-3.7210	0
0.9593	0.6931	1.2733	0.6931	1.2796	0.6931	1.0598	0.6931	0.7685	0.6931
0.0675	1.0986	0.2208	1.0986	0.0960	1.0986	0.1589	1.0986	0.5226	1.0986
-0.0335	1.3863	-0.1021	1.3863	-0.1215	1.3863	-0.1213	1.3863	-0.2302	1.3863
-0.1211	1.6094	-0.4512	1.6094	-0.2237	1.6094	-0.1828	1.6094	-0.2370	1.6094
0.2205	1.7918	-0.0646	1.7918	0.1187	1.9459	0.0570	1.7918	0.1139	1.7918
0.2069	1.9459	0.0610	1.9459	0.1469	2.0794	0.1182	1.9459	0.2332	1.9459
0.1625	2.0794	0.2207	2.0794	0.1710	2.1972	0.2356	2.0794	0.1238	2.0794
0.0446	2.1972	0.3758	2.1972	0.0601	2.3026	0.1023	2.1972	0.0554	2.1972
0.1837	2.3026	0.1145	2.3026	0.0653	2.3979	0.1366	2.3026	0.1261	2.3026
0.0303	2.3979	-0.0502	2.3979	0.1210	2.4849	-0.0443	2.3979	-0.0695	2.4849
0.1220	2.4849	-0.0450	2.4849	-0.0355	2.5649	-0.1543	2.5649	-0.1355	2.5649
-0.1802	2.5649	0.1532	2.6391	-0.0989	2.7081	-0.0540	2.7726	0.0687	2.7081
0.0883	2.7726	0.0802	2.7081	-0.1198	2.7726	0.0646	2.8332	0.0754	2.7726
0.1377	2.8332	-0.0457	2.8332	0.0506	2.8332	-0.1271	2.9444	-0.0449	2.8904
-0.0806	2.9444	0.0315	2.9957	-0.0376	2.8904	-0.0786	2.9957	-0.0604	2.9444
-0.0536	3.0445	-0.1162	3.1781	-0.0418	3.0445	-0.0530	3.0910	-0.0604	2.9957
-0.0531	3.0910	-0.0398	3.2581	-0.0425	3.1781	-0.0346	3.1355	-0.0291	3.0910
-0.0736	3.1781	-0.0551	3.2958	-0.0768	3.2581	-0.0850	3.1781	-0.0315	3.1355
-0.0926	3.3673	0.0590	3.3322	-0.0528	3.2958	-0.0516	3.2189	-0.0397	3.1781
-0.0505	3.4012	0.0698	3.4012	-0.0471	3.4012	-0.1098	3.3673	-0.0364	3.2958
-0.0535	3.4340	-0.0539	3.4657	-0.0520	3.4340	-0.0569	3.4012	-0.0513	3.3322
-0.0621	3.4657	-0.0838	3.4965	-0.1117	3.4965	-0.0398	3.4657	-0.0474	3.4340
-0.0719	3.4965	-0.0469	3.5264	-0.0750	3.5264	-0.0625	3.4965	-0.0493	3.4657
-0.0475	3.6109	-0.0679	3.5553	-0.0379	3.5835	-0.0411	3.6109	-0.0506	3.5835
-0.0363	3.6376	-0.0407	3.6636	-0.0393	3.7377	-0.0589	3.6376	-0.0449	3.6109
-0.0457	3.6636	-0.0550	3.7842	-0.0422	3.8067	-0.0400	3.8067	0.0435	4.3567
-0.0687	3.6889	-0.0639	3.8067	-0.0406	3.8918	-0.0387	3.8286	0.0590	4.3944
-0.0315	3.7136	-0.0563	3.8286	-0.0365	4.0604	0.0466	4.3175	0.0817	4.4543
-0.0317	3.8286	-0.0547	3.9318	0.0405	4.8828	0.0420	4.3438	0.1514	4.4659
-0.0424	3.9318	-0.0456	4.1431	0.0527	4.8903	0.0341	4.3567	0.1381	4.4773
0.0292	4.8122	0.0412	5.1299	0.0890	4.8978	0.0471	4.3694	0.0730	4.4886
0.0341	4.8828	0.0451	5.1358	0.0557	4.9053	0.0404	4.3944	-0.0279	4.5109
0.0296	4.9345	0.0381	5.1417	0.0420	4.9273	0.0364	4.4067	0.0454	4.5433
0.0301	4.9488	0.0278	5.1818	0.0481	4.9345	0.0372	4.4308	-0.0284	4.7707
0.0299	4.9628	0.0311	5.1930	0.0443	4.9416	0.0501	4.4427	0.0422	5.1358
0.0625	4.9698	0.0309	5.2204	0.0463	4.9488	0.0393	4.4543	0.0498	5.1417
0.0409	4.9767	0.0423	5.8289	0.0373	4.9698	0.0424	4.4773	0.0452	5.1761
0.0303	6.0684	0.0283	6.7417	0.0397	4.9904	0.0416	4.4998	-0.0401	5.1874
0.0472	6.0776	0.0301	7.3369	0.0382	5.5797	0.0356	4.5218	0.0398	5.5413

Table –I Mel cepstrum for five different speckers.

DEMONSTRATIONS

A Smart Electronic Bangla Dictionary

Shamita Ghosh, B. B. Chaudhuri
Indian Statistical Institute
203 B. T. Road, Kolkata- 700 108
shamita@rediffmail.com, bbc@isical.ac.in

A dictionary is a large collection of words, which provides spelling, pronunciation, parts of speech, meaning and other useful information. In this computer era it is possible to implement the dictionary electronically. An electronic dictionary is superior to traditional printed dictionary for different positive reasons.

In 1999, the first Bangla-Bangla electronic dictionary was constructed at CVPR Unit, ISI Kolkata. In our dictionary, there are 65,000 words taken from two printed book dictionary (Samsad Bangla Abhidhan and Akademi Vidyarthi Bangla Abhidhan). The parts of speech (grammatical category) and synonym information was also included in this electronic version. This paper describes this pioneering development, which incorporates many useful features for a wide variety of applications.

We can go to a dictionary for any pieces of information that may be temporarily or permanently missing from our own personal, mental lexicons, whether we are a speaker of a foreign language learning Bangla or a Bangla-language speaker unsure of spelling, meaning or other vocabulary item. Our software shows meaning of a word instantly when spelling is correctly known to the users.

There are problems with traditional paper dictionary for unknown spelling. Words are listed in a dictionary in alphabetic order. But if a person does not know how to spell a word, how can they find it? This needs search at several places of the traditional dictionaries but electronic Bangla dictionary have a solution. It allows to write in the spelling we know or imagine to be and its efficient algorithm fetches the phonetically similar but correct words in the dictionary. Then, one may select the desired word from the list of all phonetically similar words. For this purpose, a phonetic conversion of words is done in the original dictionary word file.

This dictionary also supports meaning for euphonetic and assimilated words (which are not available in the original printed dictionary). If the user types a word and there are no entry for that word in the dictionary file, then the software divide the words in two parts each of which has a meaning. Then, it shows the meaning and other information of each part (if available in the dictionary file). It divides the word in such a way that first part is the maximum length valid word. In this way, one can guess the meaning of a compound word although the whole word is absent in dictionary file.

Electronic Bangla dictionary has also an option to add new words with meaning and parts of speech. These new words are stored in separate entity called personal dictionary. This facility is not available in traditional paper dictionary. In this way, one can create his/her own dictionary of their choice.

We know that synonyms are very useful ways of recalling words we have forgotten. They also represent a rich resource for expanding our vocabulary and teach us to use words more precisely. For this reason, special books and databases are available with words having only synonyms. These books are called thesauruses. In electronic Bangla dictionary we can use a thesaurus-like option under this option. It is also capable to generate something like a thesaurus, so that the user may choose the appropriate synonym.

Electronic Bangla dictionary can also search for the words, which have the same rhyme that is identity of sound between the endings of words. Rhyming dictionaries help us find just the right word when we are writing verses or songs.

For any given character combination, the dictionary will automatically find words having such a combination. In this dictionary we can search for words when we do not know all the letters in them. We use the standard wildcards "?" and "*" to help in such situations. This feature can be used to make crossword puzzle dictionary. Many statistical data about language can also be calculated with the help of this feature of electronic Bangla dictionary. It can be a useful in the field of quantitative linguistics.

CATT
(Corpus Analysis Tool – Tamil)

M. Ganesan
CAS in Linguistics
Annamalai University

CATT comprises of 11 tools. These tools are useful to extract various linguistic information from Tamil corpus.

i) ***Code conversion***: It enables to convert the encoding of the text from one system to another. For example, if a text is stored in ISCII format, it can be convert to TAB coding system. It uses of conversion table, which can be created by the user.

ii) ***KWIC concordance***: It produces KWIC concordance for a given key. The key may be a word / phrase / prefix / suffix / infix / affix. The concordance list can be sorted on the key / the previous word / the following word. It provides information like i) the number of files processed ii) number of words and iii) the number of occurrences for the present key word. It also facilitates for storing and printing the concordance list.

iii) ***Word – part search***: Any part of a word can be searched on the entire corpora or selected text for its entire occurrence. The part of the word can be a prefix / infix / suffix / affix. The word list having the key will be presented in a sorted order with frequency of each word form. Total number of occurrences and the highest frequent occurrence are marked. The sorting can be done on frequency and the results can be stored and printed for future use.

iv) ***Text search***: Texts in the corpora are organized on seven-way classification. Based on the specification, any text from the corpus can be extracted. It also provided facility to add new text in the corpora.

v) ***Indexing***: Any word or part of a word can be searched on the entire corpora or on a selected text for its occurrence. It lists the occurrence of the key with the file name and frequency.

vi) ***Word list***: Word list with frequency can be created for the corpora or selected files. It also gives type / token information for the selected files.

vii) ***Reverse word list***: Word list can be prepared in the reverse order and sorting will be done on the reverse order.

viii) ***KWIC with file name***: KWIC as above appears with the file information. It facilitates for getting full context of the usage of the word

ix) ***Multiple condition search***: Word or part of word can be extracted with multiple conditions. For example, a word starting with particular stem and having a particular suffix can be searched. The result will be presented in sorted order with frequency.

x) ***Tag structure search***: It needs tagged corpus. The structure of our requirement can be given in terms of grammatical category. All the matched patterns will be extracted from the corpus. The search pattern can have "optional" and "obligatory" tags with special marking.

xi) ***Word-form search***: It also needs tagged text using this tool. For example, all the finite verbs in the corpus can be extracted with specific conditions, like finite verb with present tense markers or with third person singular masculine pronominal termination, etc. It can also be conditioned that a search can exclude a specific grammatical category.

A Generic Architecture for Morphological Generators

Uma Maheshwar Rao, G., Chaithra, T.P., Santosh Jena
University of Hyderabad
guraosh@uohyd.ernet.in/guraohyd@yahoo.com

The following are the components in the generic architecture of the generator proposed here:

1. A lexicon consisting of simple roots/stems in the language. Each entry is provided with the category information and inflection-type or paradigm type feature information.

2. A list of the categories of inflection or affixes (for ex. Tense, Aspect and Modal in the case of verbs and number, case and post-positions in case of nouns) are organized in terms of layers or strata.

3. A module consisting of Inflectional GNP endings and Particles and clitics.

4. A set of rules to concatenate the roots/stems and affixes listed in 1., 2. and 3.

5. A set of tables projecting allomorphic variants. Each table constitutes a finite number of cells predetermined by the surface realization of the lexeme in the context of a morphological category.

The proposed architecture involves the identification of different strata or layers among the affixes which enter into concatenation to generate word-forms. It allows to modify and increment the system easily, since the suffixes are arranged in various layers or strata and separated from the main program.

In the present model of morphological generator, the corpus is used to extract fully inflected wordforms. Testing is done by marking the morpheme boundaries, running them on the generator, and then veryfying the fully inflected output forms produced by the generator with the wordforms extracted from the corpus. The testing process requires identification of the root forms, paradigm type and the exact inflections on the word. In case of a mismatch, the program reveals where exactly there is a mismatch, which could either be a problem in wordform analysis or a problem with the design of the generator. A successful model is expected to exhibit an efficiency of 90% accuracy.

UNL-based Multi-lingual Question Answering Without Translation

Pushpraj Shukla, Pankaj Goyal, Kumar Kapil, Amitabha Mukerjee, Achla Raina
Indian Institute of Technology, Kanpur
Kanpur – 208016, UP, INDIA
{praj,pankajgo,kapil,amit,achla}@iitk.ac.in

We present a restricted domain, multilingual Question–Answering system, which can given source text in some language, answer queries in some other language, without translating the sources into the language of the questioner. The current implementations are for source text in English and questions posed in Hindi or English. The cross-language functionality has been achieved by converting the queries and the documents to an intermediate representation, an inter-lingua called UNL. Built under the premise of a homogeneous language-independent encoding, the UNL can be used as the predicate knowledge base on which inferences can be performed effectively.

Two major tasks are involved in the system implementation. The first is that of ***document processing*****,** which is to be done once for each document we wish to be queried. This converts the document into the interlingua, with added inference rules. The other involves ***query processing and answer extraction*** and is done once for every query. Our present work is based on an English corpus for safe drinking water built from documents obtained from the official websites of EPA and WHO [1].

Document processing deals with manually annotation and processing of the corpus through a Predicate Generator (the UNL Parser) to get the complete set of semantic predicates for the document (the "UNL expression" for the corpus). Some common known facts are also added to provide context information which would be commonly known. An inference engine then augments it with inferences obtained by applying a set of First Order Logic rules(inferences) to sets of predicates.

Every query is processed to get an answer template that represents the form of the potential answer corresponding to the question. This template is input to an HPSG Parser which outputs a pseudo UNL expression corresponding to the proposed answer template. The pseudo UNL expression is subsequently subject to a structure matching with the UNL document. The multilinguality of the system comes from the multilingual characteristic of the parser and the language – independent semantic structure provided by the inter-lingua (UNL).

Here is a sample question and its answer, along with the various stages in processing.

Source Text

"At some level, minerals are considered contaminants that can make water unsafe"

The Question(s)

जल को कौन असुरक्षित बनाता है ? (What makes water unsafe?)

The Answer Template(s)

जल को X असुरक्षित बनाता है। (X makes water unsafe.)

generated from the transformational rule(s)

1: कौन:2 > 1:X:2 (and what:V:1 > X:V:1)

The NL Semantic Predicates

agt(makes,X)	agt(*बनाता*,X)
obj(makes,water)	obj(*बनाता*, *जल*)
gol(makes,unsafe)	gol(*बनाता*, *असुरक्षित*)

The UNL expression for the query (for both English and Hindi)

agt(make(agt>thing, obj>thing). @present, X)
obj(make(agt>thing, obj>thing). @present , water(icl>liquid))
gol(make(agt>thing, obj>thing,), unsafe(aoj>thing))

obtained from UW dictionary entries like –

[जल] "water(icl>liquid)" (NOUN,SG) <HINDI>
[असुरक्षित] "unpalatable(aoj>thing)" (ADJ) <HINDI>

The Search returns –"contaminant(icl>matter)" as answer.

The De-conversion back to NL

[प्रदूषक] "contaminant(icl>matter)" (N) <HINDI>

The Answer(s)

1. जल को प्रदूषक असुरक्षित बनाता है (Contaminant makes water unsafe)

सार्थक : A Multilingual HPSG Parser

Department of Computer Science and Engineering,
Indian Institute of Technology, Kanpur
Contact: *{amit, praj}@cse.iitk.ac.in*

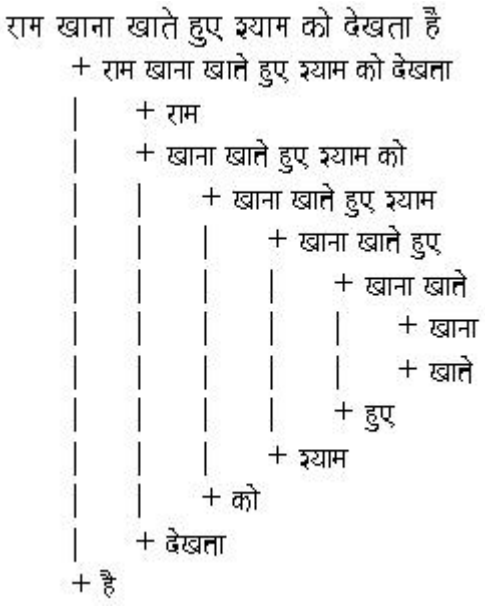
सार्थक, a generalized multilingual syntactic parser, is based on the Head-driven Phrase Structured Grammar (HPSG) framework and operates on input strings from Hindi, English, and Bengali, which are some of the structurally most disparate languages used in India. It can also handle some Hindi - English code-switching structures resulting from contact between these languages in the Indian context. The system is based on partial grammars of Hindi, English and Bengali, which we have developed using the existing HPSG formalism. The larger aim of the effort is to provide multilingual tools for animating stories created by children and semi-literate adults.

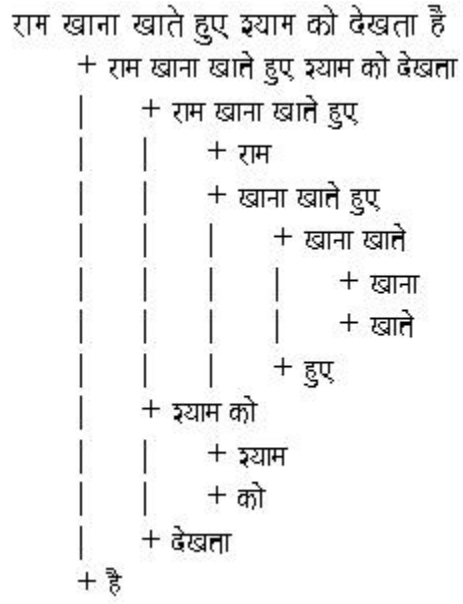
Whereas the grammatical description for English is drawn from the available work on this language , a partial HPSG grammar of Hindi and Bengali have been developed here to provide the basis for the generalized parser. The parameters such as word order and morphology on which the languages differ signifcantly are incorporated into the lexicons.

The parsing of the input involves searching for all possible phrase combinations which form a given sentence. In HPSG terms, it essentially boils down to finding the heads of all constituent phrases in a given sentence and the phrases that saturate the sub-categories of these heads. We apply a dynamic programming approach to the problem of parsing which essentially means that the main problem is broken down into sub problems. These sub problems are solved ensuring that the overlapping sub problems do not get re-solved. This procedure is called `Chart Parsing in which a chart is maintained which stores intermediate parses for constituent phrases.

With sentences that are ambiguous all possible parses are generated. Our system does not always assume a grammatically well-formed input; it is capable of recognizing sentences which are ungrammatical.

An Example : राम खाना खाते हुए श्याम को देखता है,
This is an ambiguous sentence where the the participial adjunct is attached to the phrase श्याम in one reading, and to the phrase राम in the other. The parser generates two outputs as expected. The parsed outputs are –





COMMUNICATION EMPOWERMENT LABORATORY

Indian Institute of Technology Kharagpur
INDIA 721302

Morphological Analyzer (Bengali/Hindi)

The task of the morphological analyzer is to identify the structural components of a word, and hence glean information about it. For Bengali, the morphological analyzer can identify the tense, aspect, modality and person of an inflected verb form. For Hindi, gender and number may be identified as well. Also, the root or *'dhaatu'* of the verb will be identified by the analyzer. This is performed by the morphological analyzer using a Directed Acyclic Morphological Structure, similar to word graphs and tries. For nouns, the task of the morphological analyzer is to determine its *vibhakti* (inflection), suffixes and prefixes.

Shruti: Text-to-Speech Synthesizer (Bengali/Hindi)

The building blocks of concatenative synthesizers are the signal units that are concatenated. In our cases we are using partnemes and phonemes. The units have been spliced out from prerecorded utterances taking into consideration the combination of all vowels and consonants. Given a sentence in text the synthesized speech will be generated after some steps. In the first stage it will be analyzed by a Natural Language Parser. After Morphological and Phonological analysis, the grapheme string is converted to a phoneme string, which can be directly mapped to the partneme dictionary and concatenated.

Text Editor (Bengali/Hindi) with TTS and Spell Checker (Bengali)

The text editor can be used with a wide variety of standards including ITRANS, Wx, IISCI, KGP-ISCII (a modified form of ISCII defined at Communication Empowerment Laboratory, Kharagpur). The editor can be easily reconfigured to be used with any Bengali/Hindi fonts using the keyboard mapping of the font directly. The editor is integrated with a text-to-speech synthesizer and the Bengali version of the editor is integrated with a spell checker.

Sanyog: Iconic communication system (Bengali/Hindi)

Sanyog is used as a natural language communication system for people with disabilities. It takes as input iconic sentence and produces a natural language text. The system employs a natural language generator to generate syntactically and semantically correct natural language sentence from a set of root words that are provided by the user in the form of icons. The generator in turn consists of morphological synthesizer for nouns and verbs and word planner (to order words in a sentence). Internally, the system produces an intermediate form from the input (case frame), which is then fed to the language generator to produce a natural language sentence. The text-to-speech synthesizer is integrated with the system to speak out the produced text.

Resource Center
For Indian Language Technology Solution – ORIYA

Department of Computer Science and Application,
Utkal University, Bhubaneswar, Orissa.

IMAGE PROCESSING

DIVYADRUSTI (OCR-Oriya integrated with Text-To-Speech)
English Reader System

SPEECH PROCESSING

Text-To-Speech for Oriya Language
Oriya Speech-To-Text System

NATURAL LANGUAGE PROCESSING

ORIDIC [e-Dictionary (Oriya ⇔ English ⇔ Hindi)]
ORINET (Oriya WordNet)
ORI-SPELL-CHECK (Oriya Spell Checker)
Trilingual Word Processor
Oriya Grammar Checker
Oriya Morphological Analyser
OMTRANS (Oriya Machine Translation)
Sanskrit Word-Net

Department of Computer Science & Engineering
Punjabi University, Patiala, Punjab

Prof. G. S. Lehal

OCR FOR GURUMUKHI SCRIPT

PUNJABI WORD PROCESSOR

Which has the following features:

- Punjabi spell checking
- Punjabi sorting
- Punjabi-English dictionary
- Font conversion

AUTHOR INDEX

Addendum to the lecture compendium, The National Symposium on Morphology, Phonology and Language Engineering, Simple'04,19th-21st March, 2004.

Speech Corpus for Text/Language Independent Speaker Recognition in Indian Languages

Hemant A. Patil and T.K.Basu

Abstract— Automatic speaker recognition (ASR) refers to the detection of a person's identity from his/her voice with the help of machines. ASR finds its potential applications in telephone based financial transactions, purchase of credit card, information retrieval for surveillance in defense and intelligence organization and study of different dialects in a particular language. In this paper, we describe the methodology and a typical experimental setup used for building up speech corpora for the text-independent speaker recognition in Indian Languages viz. Marathi, Bengali, Hindi, Urdu, Tamil and Telugu in the Department of Electrical Engineering, IIT Kharagpur.

Keywords: Speaker recognition, dialectal regions in Maharashtra, data collection, corpus design.

I. Introduction

AUTOMATIC Speaker Recognition (ASR) refers to the task of identifying a person from his/her voice with the help of machines. The basic difference in these lies in the number of decisions required for performing the task. For example in SID and SC, the number of decisions equals to the total number of speakers or acoustic classes respectively, stored in the machine whereas in SV, we need to compare the claimed identity with only one stored identity of the speaker [5].

The performance of ASR is dependent on database. The factors affecting the performance are recording conditions, gender type used in speaker population, etc. Success rate obtained in an ASR is meaningless if the recording conditions are not known. The use of standard speech corpora for evaluation of ASR is the most crucial task in speech and speaker recognition systems. In this paper, we describe a methodology and a typical experimental setup used for collecting speech corpora in Marathi and Hindi language from six distinct dialectal regions of Maharashtra. Data collection for Marathi, Hindi, and Urdu is completed whereas for the Bengali, Tamil and Telugu, it is in progress. This paper will emphasize on Marathi and Hindi languages. To the best of the authors' knowledge, there is no publicly available corpus in Indian languages for ASR in real life settings, so it was decided to design and develop a suitable corpus for this purpose.

Section II describes different dialectal regions of Maharashtra. Section III describes briefly a typical experimental setup, database and corpus design procedure. Section IV discusses the different salient features of the corpus. Section V discusses briefly our efforts and experiences during corpus building and finally section VI concludes the paper.

II. Distinct Dialectal Regions in Maharashtra

Fig. 1 shows the dialectal map of Maharashtra State which can be broadly classified into six major dialectal regions viz. Khandesh, Pune, Kolhapur, Konkan, Marathwada and Vidharbh where majority of the people speak Marathi as their native language and Hindi as the non-native language. There are distinct dialectal differences in the spoken form of Marathi in these regions. The factors which are responsible for these differences/deviations are diction, semantics, pronunciations, idiosyncrasies, intonation, and rhythm. One of the most important factors affecting dialectal differences is the geographical barriers creating isolation of region making intermixing of people less probable. For example, Kolhapur and Konkan region, though adjacent, are separated by *'Phonda Ghat'* and hence show different dialectal features. Table I below shows native languages spoken in each of these dialectal regions.

TABLE I
NATIVE LANGUAGES IN MAHARASHTRA

DIALECTAL REGION	NATIVE LANGUAGE
Khandesh	Ahirani
Marathwada	Marathwadi
Pune	Puneri
Kolhapur	Kolhapuri
Vidharbh	Warhadi
Konkan	Konkani

Hemant A. Patil is a research scholar in the department of electrical engineering, IIT Kharagpur, West Bengal, India.
(E-mail: hemnat@ee.iitkgp.ernet.in).

T.K. Basu is with the department of electrical engineering, IIT Kharagpur, West Bengal, India (E-mail: tkb@ee.iitkgp.ernet.in).

Fig. 1. Dialectal map of Maharashtra

III. Typical Experimental Setup at IIT Kharagpur

Fig. 2 describes a typical experimental setup developed for ASR at Speech Processing Laboratory (SPL), in the Department of Electrical Engineering, at IIT Kharagpur, India. The setup consists of a close talking microphone, voice activated tape recorder and Pentium-III machine having speech processing software. Other studies in corpus design for ASR in English language can be found in [1],[3],[4], [6-8].

A. *Data Collection and Design of Corpus*

Database of 200 speakers is prepared from the six distinct dialectal regions of Maharashtra viz. Khandesh, Pune, Kolhapur, Konkan, Marathwada and Vidharbh in Marathi and Hindi languages with the help of voice activated (VAS) tape recorders (Sanyo model no. M-1110C and Aiwa model no. JS299) with microphone input and close talking microphones (viz. *Frontech and Intex*). The data is recorded on the Sony high fidelity voice and music recording cassettes (C-90HFB). A list consisting of five questions, isolated words, digits, combination-lock phrases, read sentences and a contextual speech of approximately 90s duration was prepared in both Marathi and Hindi languages. The contextual speech consisted of description of nature or memorable events etc. of community or family life of the speaker. The topics generally provided speaker to think instantaneously and the speech was usually conversational and quite varied. The interview was started with some questions to know about the personal information of the speaker such as his name, age, education, profession, etc. After that the list was given to speaker to read in his own way. During the interview, both the speaker and interviewer were able to see and interact with each other. The data was recorded with 10 repetitions except for the contextual speech.

TABLE II
Database Description

Item	Details
No. of speakers	200 (~50%M & ~50% F)
No. of sessions	2
Intersession Interval	Hours-days-weeks
Data type	Speech
Sampling rate	22,050Hz
Sampling format	1-channel, 16bit resolution
Type of speech	Read sentences, isolated words and digits, combination-lock phrases, questions, contextual speech of approximately 90 sec duration.
Application	Text-independent ASR system
Training language	Marathi, Hindi
Testing language	Marathi, Hindi
No. of repetitions	10 except for contextual speech.
Training segments	30s, 60s, 90s, 120s.
Test segments	1s, 3s, 5s, 7s, 10s, 12s, 15s.
Microphone	Close talking microphone
Channels	EP to EP Wire
Acoustic environment	Road/home/office/slums/college/trains/hills/ valleys/remote villages/research labs/farms
Evaluation procedure	None

During recording of the contextual speech, the interviewer asked some questions to speaker in order to motivate him to speak on his chosen topic. This also helps the speaker to overcome the initial nervousness and come to his natural mode so that the acoustic characteristics of his speech are tracked precisely. The speaker's voice and interviewer's voice were recorded on the same track. After the recording was done for each speaker, a speaker information form was filled up. The form was prepared to know historical details of the speaker such as his name, age, sex, place of birth, native place, social status, education, profession, no. of children if

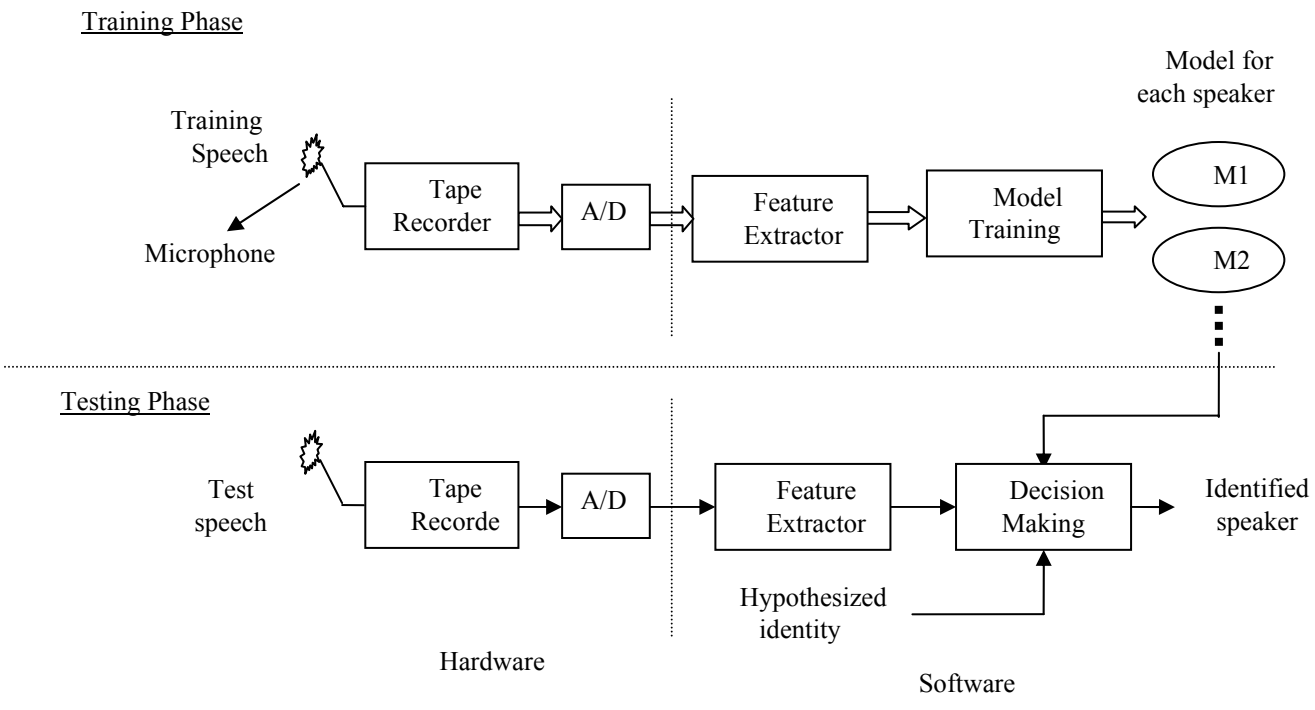


Fig. 2. Typical experimental setup for text-independent ASR system at IIT Kharagpur

married, mother tongue, languages known, etc.

B. Data Acquisition

The recorded voice is played back into the computer through the sound card of a computer (P3-833MHz, 256MB RAM). The communication between the headphone output from the VAS and sound card was made with the help of *EP-EP pin*. The speech files were stored into the '.wav' files with the help of Sound Forge Ver 5.0 sound editing software with 22050Hz sampling frequency and 16 bit depth. The dc level

adjusted into the software in order to eliminate the dc offset produced by audio hardware, i.e., when the recording hardware such as sound card adds dc offset to the voice signal, which may produce glitches and other unexpected results, if not properly compensated. Once the magnetic tape was played into the computer, the speaker's voice was played again to check the editing errors. The interviewer's voice was deleted from the speech file so that we can make the *voiceprints* (a model describing the acoustical characteristics of speaker) of the actual speaker. The automatic silence detector was employed to remove the silence periods in the speech recordings to get only the voiceprints of the speaker and not the background noise and silence interval. Also, each speaker's voice is normalized by the peak value so that the speech amplitude level is constant for all the speakers.

C. Corpus Design

Using a corpus to perform any experiment for speaker recognition requires the definition of an evaluation procedure that specifies, among the other things, the partitioning of a corpus into training and testing data sets. Examination of system performance for specific conditions such as intersession variability, microphone variability, mimic resistance based on physiological characteristics (eg., identification of Identical Twins and Triplets) and behavioral characteristics (eg., identification of professional mimics) requires that the corpus have enough speech from enough speakers for the condition of interest to construct a valid experiment. In addition, for speaker verification experiments, a corpus is required to be sufficiently large so that two different sets of speakers can be used for training and testing [2]. Corpus is designed into training segments of 30s, 60s, 90s and 120s duration and testing segments of 1s, 3s, 5s, 7s, 10s, 12s and 15s in order to find the performance of the system for various training and testing duration. Table II shows the summary of the data collection procedure and corpus plan.

IV. Features of the Corpus

Following are some of the distinct features of our corpus.

- Text was different in each session.
- Two different languages were used. So the data is completely phonetically independent.
- Corpus is suitable for both mono-lingual and cross-lingual experiments.
- Speech units including specific vocabularies e.g. isolated words, digits, combination-lock phrases, read sentences, question-answer session, and contextual speech of 90s duration with varying subjects were considered in the recording.
- Wide varieties of acoustic environments were considered during recording ranging from office-roads-train, noisy workstations, etc. which added the realistic noise (eg. crosstalk, burst, spurious noise activity) to the voice data.
- Data is not recorded in closed booth (research lab) where the speaker may not feel free.
- Speakers of wide ranging ages (6-70years) and a variety of educational background (from illiterate to university post graduates) have been considered in the corpus.

V. Efforts and Experiences During Corpus Building

Following are some of our efforts and experiences during data collection phase which may be of considerable importance in practical situations.

- In each dialectal region, there was initial enquiry from the local authorities such as 'Sarpanch' or 'Police Patil' about validity of certificates from IIT Kharagpur authorities for data collection work and usefulness of this research work.
- Initially in every village the interviewer went, a meeting was held wherein Sarpanch or Police Patil of the village used to call the native speakers and the purpose of the project was explained allaying any apprehension about the misuse of the voice documents and recordings was carried out at a central place in the village.
- Many native speakers were shy and reluctant for their voice recording. So, initially an informal discussion was carried out to overcome their nervousness and recording was done after some time.
- For recordings of the children's speech, their parents were asked to sit along with the speakers help them to overcome nervousness.
- For recordings of the illiterate speakers, i.e., those who were not able to read the text material given in list, the interviewer was helping the speakers to read or in some cases, the interviewer was reading the text and the speaker was following it.
- Some speakers were not able to speak combination-lock phrases. In such cases, an initial training was given to the speakers and then recording was done. Finally, if a speaker was not able to speak these phrases then he was asked to speak only a single digit or two digit number.
- While entering each of the dialectal regions, extensive care has been taken about the geographical conditions, culture, food habits, etc. of the regions in order to establish a close interaction with the native communities and avoid any misunderstanding between interviewer and villagers especially during the recordings of female speakers.
- While recording the speech for secondary language (i.e., Hindi) some initial resistance was experienced from a small section of speakers.

- Some speakers caught cold during recording of either first or second session. However, all of these cases were recorded in a normal fashion and no manual editing or deletion of any data was performed. This makes the problem to be applicable in realistic situations.
- Speakers occasionally became bored or distracted, and lowered their voce intensity or turned their heads away from the microphone.
- Speakers sometimes felt emotionally involved while describing personal experiences on their chosen topic and thus due to the breath bursts, frequency response of microphone was substantially lowered.
- There was stammering, laughter, throat clearing, tittering and poor articulation. All these cases were recorded in normal fashion.
- Because of differences in the availability of speakers for recording, practically it is not possible to maintain the same time of recording for all speakers rather some recordings were made in early morning, afternoon, evening whereas some even in late night.

VI. Summary and Conclusions

ASR is the use of machine to recognize a person from his voice. ASR can be used in three modes: to identify a particular person from his/her voice or to verify a person's claimed identity or to classify a person's dialectal region based on his/her similar acoustical characteristics. The methodology and a typical experimental setup for data collection, corpus design and ASR in Indian languages are presented with special emphasis on Marathi and Hindi languages. Data collection work for Bengali, Tamil and Telugu languages is in progress whereas corpus design for Urdu language is going on.

Acknowledgment

The authors of this paper would like to thank those people of Maharashtra who have given their kind support and co-operation during the data collection phase and the IIT Kharagpur authorities for their extended support for this work. They also thank to Prof. S. Sinha of IIT Kharagpur for his help and suggestions in this work.

References

[1] L.M. Arslan and J.H.L. Hansen, "Language accent classification in American English," *Speech Communication*, vol. 18, pp. 353-367, 1996.

[2] J. P. Campbell, Jr. and D. A. Reynolds, "Corpora for the evaluation of speaker recognition systems," *Proc. Int. Conf. Acoustics, Speech and Signal Processing, ICASSP'99,* Vol. 2 , 15-19, pp. 829-832, March 1999.

[3] C. G. Clopper *et al.*, "A multi-talker dialect corpus of spoken American English: An initial report," Research *on Spoken Language Processing, Progress Report,* No. 24, pp. 409-413, 2000.

[4] G. R. Doddington, M. A. Przybocki, A. F. Martin and D. A. Reynolds, "The NIST Speaker Recognition Evaluation- Overview, Methodology Systems, Results, Perspective," Speech *Communication*, Vol. 31, pp. 225-254, 2000.

[5] S. Furui, "Recent Advances in Speaker Recognition," *Pattern Recognition Letters*, Vol 18, pp. 859-872, 1997.

[6] J. J. Godfrey, E. C. Holliman and J. Mcdaniel, "Switchboard: Telephone Speech Corpus for Research and Development," *Proc. ICASSP' 92*, Vol. 1, pp. 517-520, March 1992.

[7] H. He *et al.,* "European speech database for telephone applications," *Proc. of ICASSP,* Vol. 3, pp. 1771-1774, Apr. 21-24, 1997.

[8] A. F. Martin and M.A. Przybocki, "The NIST Speaker Recognition Evaluations: 1996-2001," *A Speaker Odyssey, A Speaker Recognition Workshop,* Dec. 2001.