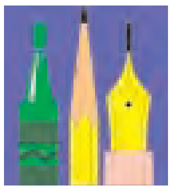
5.3 Knowledge Tools (KT)



a) Lexical Tool

5.3.1 Application Of Multilayer Perceptron Network For Tagging Parts-Of-Speech

Ahmed, S.Bapi Raju, Pammi V.S. Chandrasekhar, M.Krishna Prasad, *Language Engineering Conference, University of Hyderabad, India, Dec. 2002.*

Abstract

This paper presents a neural network based part-of speech tagger that learns to assign correct part-of-speech tags to the words in a sentence. A multi layer perceptron (MLP) network with three-layers is used. The MLP-tagger is trained with error back-propagation learning algorithm. The representation scheme for the input and output of the network is adopted from Ma et al. [6]. The tagger is trained on SUSANNE English tagged-corpus consisting of 156,622 words. The MLP-tagger is trained using 85% of the corpus. Based on the tag mappings learned, the MLP-tagger demonstrated an accuracy of 90.04% on test data that also included words unseen during the training. Results from our experiments suggest that the MLP-tagger combined with the representation scheme adopted here could be a better substitute for traditional tagging approaches. This method shows promise for addressing parts-of-speech tagging problem for Indian language text considering the fact that most of the Indian language corpora, especially tagged ones, are still considerably small in size.

5.3.2 Morphological Generator For Tamil

P. Anandan,Dr. Ranjani Parthasarathi & Dr. T.V. Geetha, *Tamil Inayam, Malaysia 2001.*

Abstract

Tamil is a relatively free word order language, the only constraint being that normally the verb comes at the end. This flexibility in word ordering is possible due to the morphologically rich nature of the language. Information such as case rules and auxiliary verbs indicating aspect, tense, mood are all conveyed through morphological attachments to the root, noun or verb. This makes morphological generation of Tamil words a challenging task. Another issue is that unlike English where number matching between noun and verb alone is necessary, in Tamil as in many other Indian languages person and gender matching between subject and verb is also necessary. A morphological generator designed for Tamil needs to tackle the different syntactic categories such as nouns, verbs, postpositions, adjectives, adverbs etc. separately, since the addition of morphological constituents to each of these syntactic categories depends on different types of information.

In this work a morphological generator has been designed for each of the syntactic categories and then combined to morphologically generate a complete sentence. The underlying morphological structure for a Tamil noun is as follows: plural suffix - if any, oblique suffix - if applicable, the euphonic suffix - if optional plus the case suffix. While generating the noun derivatives from the roots linguistic rules determining the form of a plural suffix has to be considered. The attachment of case suffixes to nouns is an important part of the morphological generator for nouns. In addition, this has to take into consideration the fact that certain nouns can take case suffixes only in oblique form. The euphonic suffix sometimes comes along with oblique suffixes or with plural suffixes. This has also to be considered. During the combination of the root - noun with the above mentioned suffixes, “sandhi rules” have to be taken in to account.

The Morphological structure of Tamil verb is quite complex since it caters to person, gender, and number markings and also combines with auxiliaries that indicate aspect, mood, causation, attitude etc. While morphologically generating the verb, the gender, number and person of the subject is necessary in order to select the appropriate suffix catering to the selected tense. The linguistic rules determining the combination of auxiliaries with the verb is also quite challenging since more than one auxiliary can attach itself to the verb as suffixes. The combination becomes exponentially large, hence we have used semantic based heuristics to prune the combinations. While using auxiliaries zit is the last auxiliary that matches with the person, number, gender of the subject and not the root verb of the combination. Similar linguistic rules are used to generate derivatives for other syntactic categories also. Thus a Tamil morphological generator for different syntactic categories has been designed and implemented. This morphological generator can be used for *suggestion list generation of a spell checker* and as *a basis for English to Tamil translation*.

5.3.3 Computer Parsing Of Bangla Verbs

Chaudhuri, B.B., N.S. Dash and P.K. Kundu *(1997) Linguistics Today 1(1):64-86.*

Abstract

This paper deals with automatic parsing of Bangla verbs. It includes identification of verbs in texts, define their roots and suffixes, analyze their declensions, detect their meaning, and determine their role in sentence. Roots are divided into 27 sub-groups according to their structure while suffixes are sub-

grouped depending on their conjoining with particular root. Grammatical properties are encoded in 1 byte of binary string to tag with suffix. Entries in root lexicon are tagged with information about their form and correspondence with suffix. When a verb form is encountered, a stripping algorithm is invoked to find root-suffix pairs. Next, a Boolean mapping algorithm is used for valid concatenation (grammatical mapping) between two parts. If a valid parse is obtained the result is presented by decoding encoded information attached to root and suffix. For robustness and accuracy the process is run on a large collection of verbs compiled from a corpus proportionally chosen from published documents between 1981-1995. Information may be used for analysis of contexts in sentences.

5.3.4 A Gurmukhi Collation Algorithm

G S Lehal, *communicated to International Journal for Communication*

Abstract

Sorting and Indexing is one of the basic necessities of the database management system. But unfortunately there does not exist any software for automatic sorting of Gurmukhi words. The collating sequence provided by UNICODE or ISCII is not adequate, as it is not compatible with the traditional sorting for Gurmukhi words. In Gurmukhi unlike English, consonants and vowels have different priorities in sorting. Words are sorted by taking the consonant's order as the first consideration and then the associated vowel's order as the second consideration. In addition the properly sorting "characters" in Gurmukhi often requires treating multiple (two or three) code points as a single sorting element. Thus we cannot depend on character encoding order to get correct sorting instead we have to develop using sorting rules of Gurmukhi linguistic collation function which convert the word into some intermediate form for sorting. The Gurmukhi collation algorithm developed takes care of following two main factors into consideration:

1. Deciding the collating sequence.

2. Taking care of primary, secondary and tertiary weight levels to be assigned to some characters based on alphabetic ordering, semi-vowel ordering and ignorable character based ordering respectively.

The collating sequence for Gurmukhi characters has been developed after discussions with linguists and studying in detail the alphabetic order in Punjabi dictionaries. The collating sequence takes care of vowels, consonants and conjunct consonants. For assigning the secondary and tertiary weight levels, semi-vowels and ignorable characters such as hyphens have to be taken care of. Since we have multiple-level comparisons to be carried out, so first the text to be sorted is transformed into a collation element table and then into equivalent sort keys. These sort keys might consist of a string of base weights followed by strings for weights used for secondary and tertiary differences. These keys can then sorted based on some sorting algorithm.

5.3.5 Enhanced Version Of Morphological Analyzer And Parser For Tamil Language

U. Madhupriya, *MCA Project, Anna University, Chennai.*

Abstract

Natural Language Processing is one of the central domains of investigation in artificial intelligence. Crucial to the progress in this domain is a better understanding of the properties of natural languages and the development of linguistic formalisms both for analyzing these properties and for computer applications.

Morphology can be defined as an internal structure of words. A Morphological analyzer breaks a word into its root word and associated morphemes. A morpheme is defined as the smallest part of a language that can be regularly assigned a meaning.

Tamil is a morphologically rich language in which most of the morphemes coordinate with the root words in the form of suffixes. Person, gender and number markings of the subject of the sentence. In addition auxiliaries, which convey modal, aspect, etc. also combine with the main verb and form a cohesive unit. Unlike most other languages, in Tamil, case markers occur along with the nouns and number markings. Hence a tool for morphological analysis of Tamil language is necessary.

Most of the languages of the world allow considerably more variation in word order than does English. The subject and object are identified not by their positions but by their inflectional endings. Parsing of Tamil sentences is a requisite for various applications. The syntactic correctness of the construct can be known from parsing. In a parser, morphological analysis of words is an important prerequisite for syntactic analysis. Properties of a word the parser needs to know are its part-of-speech category and the morpho syntactic information encoded in the particular word form.

The morphological analyzer and parser are built using Jlex – a lexical analyzing tool and java Cup – a parsing tool.

5.3.6 Latent Semantic Analysis And Applications For Tamil Documents

G. Gowri Mani, *MCA Project, Anna University, Chennai.*

Abstract

Implementing latent semantic indexing and various applications using LSI for Tamil documents is the main goal of the project. Here, two different applications are considered for the purpose. They are Information Retrieval using LSA and Intelligent essay assessor.

Information retrieval is aimed at automatically retrieving information, based on the semantic similarity of the documents, rather than on the perfect word match. Unlike most of the information retrieval systems of present, this system is aimed at retrieving document, which are conceptually similar. The system is developed as a web-enabled product, enabling any user to readily download the product from the net and use it to the fullest. The system automatically, extracts the higher order associations from the given document. Further it converts them to a semantic space, mathematically retrieving the conceptual details of the document. This semantic space is used to retrieve the related document, based on a user query.

The Logical sub modules involved are,

- Concept computing of given documents
- Query processing and document retrieval
- Updation of Documents

The Intelligent Essay Assessor (IEA) is a software tool for scoring the quality of essay content. The IEA uses Latent Semantic Analysis, which is both a computational model of human knowledge representation and a method for extracting semantic similarity of words and passages from text. Simulations of psycholinguistic phenomena show that LSA reflects similarities of human meaning effectively. This system automates the concept gathering of any given corpus. It further is expected to evaluate the essay as consistent and efficient as that of a human. It applies various strategies to arrive at the final score of any given essay.

The logical sub modules involved are,

I. Concept computing of the given corpus

II. Scoring the essay based on various criterion involved.

5.3.7 A Grammar Tool For Sentence Generation And Cross-Language Communication

P.V.S Rao, *describing work done at the CSR Lab., Virtual conference WWDU 2002 – the 6th International Scientific Conference on Work With Display Units – May 23, 2002, at Berchtesgaden – Germany*

Abstract

This paper describes a grammar tool which can be a literacy aid as well as a facilitator of cross language communication at the national level (for multilingual countries) and at the global level. It enables a user to generate:

a) fairly complex sentences in a language that he is barely familiar with, and

b) equivalent sentences in a second language with which the user need not even be familiar with.

He can incrementally convey the intended 'concept' (underlying the sentence) to the machine in a non-sentential form. This is internally represented as a structure, which is not language-specific. This structure can be converted (using appropriate grammar rules) into sentences of a language.

The tool can be used in two modes:

a) as an aid for sentence generation in real life or

as a training tool for gaining competence, either in the first language of the user or in a second language.

5.3.8 Anaphora Resolution For Malayalam And Hindi

L.Sobha, *Unpublished Doctoral dissertation submitted to MG University, Kottayam.*

Abstract

It is well known that natural languages contain anaphoric expressions, gaps and elliptical constructions of various kinds and that understanding of natural languages involves assignment of interpretations to these elements. Therefore it is only to be expected that natural language understanding systems must have the necessary mechanism for the same. Most of the anaphora resolution systems developed so far seem, to the best of our knowledge, to be monolingual, i.e., the systems are language specific and it has not always been shown that these are easily extendable to deal with other languages. VASISTH (the anaphora resolution system), in contrast, is a multilingual system, which presently handles two languages from two different language families: Malayalam, from, Indo-Dravidian and Hindi from Indo-Aryan. It can easily be extended to handle other Indian languages as well, more generally, other morphologically rich languages. What further distinguishes VASISTH from other similar systems is that exploiting the morphological richness of the Indian languages, it makes limited use of grammatical rules and uses only morphological markings to identify subject, object, clause etc. It uses limited parsing:

the information required from the parser is limited to parts of speech tagging, clause identification, subject of the clauses and person-number-gender of the NPs. Initially VASISTH was developed and tested for Malayalam, and then modified for Hindi. It is well known that pronouns often have more than one possible antecedent: the pronoun resolution mechanism of this system captures the ambiguity but does not resolve it. The system aims to resolve (and achieves considerable success in doing so —- complete success as far as reflexives, reciprocals, and distributives — are concerned) the antecedent problem of referentially dependent elements such as pronouns — both anaphoric and cataphoric uses —, including the so-called "one – pronouns", reflexives, emphatic and non-emphatic, reciprocals, and distributives, gaps, and certain kinds of ellipsis. The parser and the anaphora resolution system work in "c".

5.3.9 VASISTH An Anaphora Resolution System For Indian Languages

Sobha,L and B.N.Patnaik *(2000) International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications ACIDCA'2000, Monastir, Tunisia.*

Abstract

The paper presents "vasisth", an anaphora and ellipsis resolution system, developed originally for Malayalam, an Indo-Dravidian language, and then tested for Hindi, an Indo-Aryan language. The testing was done to see if the system would apply without modification to another Indian language, structurally similar to Malayalam in many crucial respects, and if modifications are needed, what these modifications specifically are. The test gave encouraging results: only a very few minor modifications are needed for the system to apply equally efficiently to Hindi. One fact about the computational grammar that is implemented here which deserves attention is that it does not use more sophisticated notions of modern formal linguistics, and achieves its goal with very familiar concepts such as clause, subject, object, etc., which are identified with the help of morphological information, and concepts such as precede and follow. The more complex notion of hierarchy is not used. The result is quite encouraging: a very simple grammar, from the point of view of implementation, and the coverage is not affected. The algorithm developed here works on a partial parser, which is because the need for a complete parser has been eliminated for the operation of the system. The system works with a fairly high degree of success and it can be said with some confidence that it can be extended to other morphologically rich languages.

5.3.10 Vasisth An Ellipsis Resolution In Malayalam And Hindi

Sobha, L and B.N. Patnaik. *(2000), MT 2000 – Machine Translation and Multilingual Applications in the New Millennium, University of Exeter, United Kingdom: 19-22 November 2000*

Abstract

The paper presents an algorithm which resolves elliptical constructions for Malayalam, an Indo-Dravidian language, and then tested for Hindi, an Indo-Aryan language. The algorithm is a part of an anaphora resolution system called VASISTH. The testing was done to see if the system would apply without modification to another language, structurally similar to Malayalam in many crucial respects. The test yielded encouraging results. The computational grammar implemented here uses very familiar concepts such as clause, subject, object etc., which are identified with the help of morphological information, and concepts such as precede and follow. The algorithm works on partial parser.

5.3.11 Vasisth An One-Pronoun Resolution Algorithm For Indian Languages

Sobha, L and B. N. Patnaik. *(2000), Third International Conference on Discourse Anaphora and Anaphor Resolution (DAARC2000), Lancaster University, United Kingdom 16 – 18, November 2000*

Abstract

The paper presents an algorithm, which resolves one-pronoun construction for Malayalam, an Indo-Dravidian language, and then tests it on Hindi, an Indo-Aryan language. The algorithm is a part of an anaphora resolution system called VASISTH. The testing was done to see if the system would apply without modification to another language, structurally similar to Malayalam in many crucial respects. The test yielded encouraging results. The computational grammar implemented here uses very familiar concepts such as clause, subject, object etc., which are identified with the help of morphological information, and concepts such as precede and follow. The algorithm works on partial parser.

5.3.12 Object Oriented Design Of Semantically Driven Tag Based Hindi Sentence Analyzer

Om Vikas, *Conference on Computer Processing of Asian, Languages (CPAL-2, March 1992*

Abstract

The paper discusses the design methodology of a semantically driven TAG (Trea Adjoining Grammar) based sentence analyzer which analyzes simple and complex sentences, and performs syntactic and semantic checks having access to a Semantic Relational Lexicon to facilitate semantically driven parsing. Sentence is an expression of an event or a situation that an action (A) takes place on some entities (E) which may play a role of agent, object, instrument, etc. Five E:A relationships have been specified. Five syntactic classes have been included in this paper. EP and AP are termed as Entity Phrase and Action Phrase respectively. Class I: AP-Identification Procedures (+Modality), Class II: EP-Identification Procedures, Class III: E-Inflexion Procedures, Class IV: A-Inflexion Procedures, Class V: E-A Agreement Procedures. Tree Adjoining Grammar (TAG) theory may be used for combining sentences and clauses. EP may consist of EP and the J_Phrase (starting with 'jo', 'jisne', etc). To form ('kyaa') question, K_phrase may be added in the beginning or K_phrase (starting with 'kyaa', 'kisko', 'kaise', etc.) may replace EP/AP. Two sentences may be linked to an appropriate Epi of the main sentence through L_par that is linking parsarg (e.g. 'ki', 'kyonki', etc...). The proposed analyzer analyzes a sentence into entities (E), actions (A) and their modifiers with their features. The case relations are assigned from left to right in a sentence by using six rules of identifying Agent, Object, Instrument, Goal, Source and Locus (time, situation, manner).

b) Utilities

5.3.13 Bridging The Gap Between Home Dialect And School Language Through Multimedia

N. Anbarasan, *National seminar on Research and Innovations on Home and school language issues.*

Abstract

Language learning is a complex and time-consuming process. When it comes to Language learning pupil experience the difference between Home dialect and the school language at various linguistic levels such as phonological, grammatical, lexical and semantic. The dialect is acquired at out and out through the socialization process, where as school language is learnt in a formal schooling situation.

A homogeneous classroom is not guaranteed as pupils coming from different corners having diversity in language, culture, socio economic, tradition, custom, caste, religion etc. With this background, pupils respond and re-act differently to the same situation. Multimedia helps in preparing an unambiguous material. As different individuals require different types of teaching and learning suitable to their individualism, multimedia has the scope for configuring to meet these requirements. This would ensure better results and expected outcome. Multimedia helps one to get conformed with the pace of progress, one maintains and to accelerate the rate of achievement and ensure proper attainment.

In any formal language learning/teaching situation, there is a necessity to have supplementary materials with a view to :

(1) Re-inforcement of language learning

(2) Coverage of certain aspects which have not otherwise covered in the conventional material

(3) Supply additional information/teaching inputs

(4) Overcome certain deficiencies in teaching as well as materials.

The multimedia based Teaching/Learning materials (Aids) bring life and helps pupils better understand the abstract difficult concepts easier and grasp. It also motivates the learning interest of the pupils with the real life animations with sound effects and music in it and helps reduce the labour of the teachers. Multimedia also enables the learners in re-inforcement of the subject by means of games and in turn paves ways for debates and discussions.

5.3.14 Constraints In Developing Language Software

N. Anbarasan, *Seminar International Seminar on Tamil Computing, Chennai.*

Abstract

When the computers are penetrating deep into every field of science and society, it quietly shows up its English face and puts the vernacular user to a great disadvantageous position and are left out. It happens due to the limitations of the existing operating systems, tools and application software developed.

Any software is meant for processing of the given data from some source and produce some results. It also applies to any language software. The three components of a program namely Input, Processing and Output, normally developed in a complicated way due to the complicated standards apart from the complexities of the languages themselves.

Developing software to take care of inputting Indian languages is a complex process in the requirements to develop it for different kinds of users namely Typist, Non-Typist, Casual and Professional users, all of them need different types of input methods.

The language software being developed as a sit-over the software developed for international user meant for English. These software makes uses of certain code points for their internal use Such as soft-hyphen, Soft carriage return, Paragraph marker, End of file marker etc. When the language software uses the ANSI encoding, which has been maintained over a period of time uses those codes which is used internally by the English software resulting in non-available glyphs. This leads to serious disadvantages to the language software developers as they can not develop any language software using those standard English software development tool.

5.3.15 Mother Tongue Learning Through Multimedia

N. Anbarasan, *Seminar EMMIT 98- SAARC Conference on Multilingual Multimedia Information Technology.*

Abstract

We have been using print, audio and video media for language teaching. Each of the medium has its own merits and demerits. These conventional media can be put together with a view to develop all the objective prescribed, elaborately and minutely, in the multimedia package.

The interaction between a student and a teacher generally enhances learning. In a conventional class room the interaction for all practical reasons generally is not taking place because of reasons such as psychological barrier between teacher and taught, the lack of confidence in the presence of other students, lack of privacy, lack of individual attention by the teacher etc.

As privacy is maintained, the student can interact and get all kinds of information towards learning, even below average student without any hesitation can interact and respond with any kind of answers. In this way, the learner tries to get clarified all his/her doubts which enhances his/her learning. In other words, the learning process through the multimedia package is strikingly different from the conventional classroom teaching learning method.

The multimedia package motivates the learners in participating in the active learning process. Because of the active participation of the learner, his/her mental faculty is always alert and in this process the learning is effective and generally much faster. Depending upon the requirement, a learner can choose what he wants and much more than what he requires.

The kind of learning activities provided in the package enhances the creativity of learners. A computer, unlike a human teacher never gets tired in providing all requested information's, which are stored.

The multimedia package has an inbuilt facility to evaluate the responses of the learners, and provides remedial measures for correction and further learning.

5.3.16 The Enabling Technology For Indian Languages

N. Anbarasan, *Seminar Tamil Internet 2000-Directions to the digital media*

Abstract

As far as administration is concerned, there are hundreds of general-purpose softwares available Off-the shelf. These softwares range from simple word processing to complex database management through fascinating Desk Top Publishing. These softwares are constantly revised and upgraded to keep pace with the rapidly advancing technologies in hardware and operating systems. With the ever-increasing demand, the softwares are becoming more sophisticated.

For any user, the natural choice could be to use the same English software for vernacular usage as well. As the user is already familiar with the English software and its operational details, it is convenient to have the same, rather than re-learning a new set of command. Based on this approach, certain software is being developed and it is such software, which is successfully used for the obvious reasons. This type of development could be classified as "Interface software".

The interface software merely allows inputting of Language text into the application softwares on popular Operating Systems. The formal approach could be to enable the input and display of Language text at all levels. Such software can be conveniently termed as "Enabling" software.

The software developed for English cannot be used for Indian languages, which have complex scripts. The software is developed with features specific to English-such as "Find" and "Replace", "Spell check", "Dictionary", "Auto correct", "Mail merge", "Sort", "Index" etc. The softwares available in the market to meet the requirement of vernacular demand provides no support to these features either directly or indirectly.

5.3.17 Generating Converters between Fonts Semi-automatically

Bharati, Akshar, Nisha Sangal, Vineet Chaitanya, Amba P Kulkarni, and Rajeev Sangal, *Proc. of SAARC conference on Multi-lingual and Multi-media Information Technology, CDAC, Pune, 1-4 Sept. 1998b.*

Abstract

It is important for us to be able to view as well perform search and other operations on texts in Indian languages available over the world wide web (or floppies or CDs), independent of the hardware or software platform. There are problems in doing this because currently most of the sites on Indian language texts are not following any coding standards. While the long term answer might be for everyone to switch over to a standard alphabetic coding scheme (ACII), some tools have been developed for immediate use that allow texts in glyph coding to be converted to the standard, automatically or semi-automatically.

In this paper, a system is described which takes: (i) a text in an unknown coding scheme, and (ii) the same text in the ACII coding scheme, and generates a converter between the given unknown coding scheme and ACII. The converter can be used to convert a text from the non-standard coding scheme to ACII, and back. They can be progressively refined, manually.

For generating the converter, a glyph-grammar for the script of the language is also needed, which specifies what possible glyph sequences make up an akshara. The grammar is independent of the coding schemes, and is structured. It needs to be developed only once, for a script.

A sample glyph grammar for Devanagari has been described. A system has been implemented for Devanagari script, using which converters for at least two sample scripts have been generated semi-automatically.

5.3.18 ISCII Plugin For Displaying Indian Language Web Pages

Bharati, Akshar, Vineet Chaitanya, Amba P. Kulkarni, *LTRC, IIIT, Hyderabad, 2002 (presented at Caturang, 5-6 March 2001, IUCAA, PUNE)*

Abstract

There is a chaos as far as the Indian languages in electronic form are concerned. Neither can one exchange the notes in Indian languages as conveniently as in English language, nor can one perform search on texts in Indian languages available over the web. This is so because the texts are being stored in font dependent glyph codes.

Though on the face of it UNICODE appears to be an ideal solution for this chaos, on closer look one finds that UNICODE with UTF-8 the transmission cost for Indian languages will be three times that of English! So our suggestion is till a satisfactory solution is found for overcoming the transmission inefficiency one should stick with ISCII standard. It has an additional advantage of having uniform code for all the Indian languages. This makes the conversion among Indian languages as simple as selection of font.

An ISCII plug-in has been developed to enable the display of ISCII web pages on client machines using available local fonts an ISCII plug-in has been developed. Storage of web pages in ISCII also solves the search problem.

Since the plug-in is available “free” under GPL, computer dealers can provide it as a pre-installed software in PCs. Also experts can improve upon it by adding new features to it.

Same plug-in can also be used for UNICODE with minor modifications.

5.3.19 Language Technology Solutions In Simputer : An Overview

Kalika Bali, Ramesh Hariharan, Swami Manohar, V. Vinay and K. S. Vivek, *Language Engineering Conference, University of Hyderabad, India, Dec. 2002.*

Abstract

Simputer is a low-cost multilingual, mass access handheld device that uses Indian Language User Interfaces to provide Information Technology based services to the multilingual population of India. In this paper we will discuss the language technology components of the Simputer that help realize this experience. The architecture and detailed working of multilingual text rendering and display as well as multilingual text-to-speech systems, in particular the IML browser and Dhvani TTS engine, deployed in the Simputer will be outlined.

5.3.20 Towards Indian Language Spell-Checker Design

Bidyut Baran Chaudhuri, *Language Engineering Conference, University of Hyderabad, India, Dec. 2002.*

Abstract

This Paper deals with the development of the spell-checker in Indian Languages with an example in Bangla, the second most popular language in Indian Subcontinent. A brief review of problems and current scenario of Indian language spell-checkers is described. Then the approach on Bangla spell-checker is elaborated. In this approach the technique works in two stages .The first stage takes care of phonetic similarity error. For that the phonetically similar characters are mapped into single units of character code. A new dictionary D_c is constructed with this reduced set of alphabet. A phonetically similar but wrongly spelt word can be easily corrected using this dictionary. The second stage takes care of errors other than phonetic similarity. Here wrongly spelt word S of n characters is searched in the dictionary D_c. If S is a nonword, its first k_1£ n characters will match with a valid word in D_c. (if k_1 = n then word in D_c must be longer than n). A reversed word dictionary D_r is also generated where the characters of the word are maintained in a reversed order. If the last k_2 characters of S match with a word in D_r then, for single error, it is located within the intersection region of first k_1 + 1 and last k_2 + 1 characters of S. We observed that this region is very small compared to word length for most cases and the number of suggested correct words can be drastically reduced using this information. We have used our approach in correcting Bangla text, where the problem of inflection is tackled by a simplified version of morphological analyzer. Another problem encountered in Indian languages is the existence of large number of compound words formed by Euphony and Assimilation. The problem of compound words is also carefully tackled.

5.3.21 Design And Implementation Of A Spell Checker For Assamese

Monisha Das S. Borgohain Juli Gogoi & S. B. Nair.

Abstract

Spell Checkers form a vital ingredient of text processors, character recognition systems, dictionary search engines, language processing software and similar tools. Though considerable work has been done in the area for English and related languages, the Indian language scenario present a relatively more complex and uphill task. This paper describes strategies involved in the implementation of a Spell checker for Assamese, the official language of the North Eastern Indian State of Assam.

5.3.22 Knowledge Representation For Web Based Services In A Multi-Cultural Environment

Hiranmay Ghosh, N. Rajarathnam and Santanu Chaudhury, *Proceedings of the 3rd International Workshop on Web Site Evolution (WSE 2001), Florence (Italy), IEEE Computer Society Press, November, 2001.*

Abstract

Internet being a global resource, web based applications need to break the barriers of language and culture. The core of an intelligent web based application comprises an ontological description of the domain. A domain ontology needs a medium for expression, which usually consists of terminology borrowed from a natural language. Thus, a knowledge-based application becomes susceptible to linguistic and cultural context. In this paper, we present a new knowledge representation technique that distinguishes between the abstract concepts in a domain and their expressions. It can associate expressions from different languages with the concepts in an ontology network. Non-textual symbols and media property specifications can also be used to express the concepts using this technique. The resulting ontology can thus be used in a multi-lingual and multi-cultural environment. An RDF based language is used as a vehicle for the knowledge representation scheme.

5.3.23 Malayalam Speech Sounds And Their Mapping To Unicode Symbols: A Case Study

Dr.V.Geetha Kumary and Dr.B.A.Sharada, *Language Engineering Conference, University of Hyderabad, India, Dec. 2002.*

Abstract

This paper deals with the problems in the representation of phonetic symbols of Malayalam language in html page. These symbols are not the ones normally available on the computer keyboard. In many cases it used to be written by hand and only few symbols were available in the insert mode in the word processors and many times they were not compatible with html packages. This paper focuses on the phonemes and allophones of Malayalam and their transliteration and presentation on the web. The outcome of the study is a table consisting of the Malayalam phonetic symbols with their respective Unicode numbers and an explanation of utility. This study is helpful in the usage of Unicode in representing Indian language fonts at all levels such as in language teaching materials, in creating web documents in descriptive linguistics, and in converting the printed documents to digital resources.

5.3.24 Text Mining In Request For Comments Document Series

Siva Guruswamy, D.Manjula, T.V.Geetha, *Language Engineering Conference, University of Hyderabad, India, Dec. 2002.*

Abstract

This paper discusses the knowledge discovery in text (KDT) system for the ‘Request for comments (RFC) Document Series’. The paper proposes the versatile system architecture for the Text Mining in RFC that maintains structured and unstructured data

components of the document. The documents are represented by keywords and knowledge discovery is performed by analyzing the co-occurrence frequencies of the various keywords representing the document. The clustering of the documents is done by extracted knowledge, which can reduce the search space for searching. The relevant documents retrieved during the search process for a query are ranked based on the relevance of topic in it. This paper describes RFC Viewer, our tool for viewing the RFC document in rich text format rather than in text format, it also provides the knowledge extracted from the RFC document and supports various KDD Operations on the document.

5.3.25 Spellchecker for Tamil

M.V.Hemalatha, *MCA Project, Anna University, Chennai.*

Abstract

The aim of this project is to provide a Spellchecker for the use of a Word processor in Tamil. The Spellchecker will be invoked by the Word processor word by word. Spellchecker takes a Tamil word from the Word processor as its input and determines whether it is correct or incorrect. If the word is misspelt, it tries to correct the word and generate possible suggestions. The Spellchecker makes use of the Morphological analyzer for splitting the given Tamil word into root word and a set of suffixes and Morphological generator to generate the suggestion list from the root word and a set of suffixes. The Spellchecker is implemented in JAVA 2.

5.3.26 Kids Tutor

A.Kumaran, *MCA Project, Anna University, Chennai.*

Abstract

Kids Tutor is developed for school children of age 7 to 13. The project was developed in a way to make learning a fun. This covers four modules namely – Word Hunt module, Narrator module, Reactor module and Abacus module. This project is developed using Java.

Word Hunt module is so designed, that with its help, the “intelligent quotient” (IQ) of school children can be tested. Pictures of categories like animals, fruits, vegetables etc (flexibility of adding new category is also provided according to one’s choice) will be shown. The student has to identify what is in that picture and identify it in the words grid provided.

Narrator module deals with stories. This was developed with the idea of improving the listening skills, the reading ability and also vocabulary of students. The story is narrated along with images running along its side.

Reactor module is for students of class six and above. This deals with identification of elements and radicals along with their symbols and valencies. It also deals with finding the correct chemical equations. This module has objective type based questions, for which choices will be given. The student has to choose the right answer for which marks will be awarded.

Abacus module deals with the numerical ability of the students. This covers simple addition, subtraction and the concept of fractions with the aid of images. A game in simple arithmetic is also provided. This tests the mathematical ability of the students and improves their alertness and interest towards mathematics.

5.3.27 Semantic Based Text Mining

D. Manjula, P. Malliga and T.V. Geetha., *1st International Global Wordnet Conference, CIIL, Mysore, Jan 21-25,2002.*

Abstract

This paper discusses the incorporation of semantics in the various phases of Text Mining. Text mining is the process of extracting implicit, previously unknown and potentially useful information from textual documents. Domain Concept is also needed in information extraction because document collections are in linguistic, domain specific and application levels. The WordNet is a lexical database, which does not have the domain knowledge in detail. The domain knowledge is introduced in the process of text mining to improve the retrieval performance. The interlinked domain concept trees are to be created for the domain knowledge.

5.3.28 Web Based System For Teaching Tamil Grammar

Pugalendhi.V, *MCA Project, Anna University, Chennai.*

Abstract

The aim of this project is to provide a web-based system for teaching Tamil grammar. This web based system for teaching Tamil grammar is classified into four distinct modules namely web pages to teach Tamil grammar, online games, online exam and picture-voice module.

The web page is to teach grammar module consists of two parts. The first part consists of five categories, each dealing with a specific area of Tamil grammar. The second teaches the Tamil script. It contains pages with animated to Tamil an alphabet that explains how

these alphabets are written. The online game module consists of four games that deal with vowels, consonants, nouns, verbs and building words. The online grammar exam module deals with generating question paper for the online exam. It also provides an interface to maintain the question bank. The picture-voice module helps in pronouncing Tamil words and knowing its meaning.

5.3.29 An Architecture For A Text Simplification System

Advaith Siddharthan, *Language Engineering Conference, University of Hyderabad, India, Dec. 2002.*

Abstract

We present a pipelined architecture for a text simplification system and describe our implementation of three stages—analysis, transformation and regeneration. Our architecture allows each component to be developed and evaluated independently. We lay particular emphasis on the discourse level aspects of syntactic simplification as these are crucial to the process and have not been dealt with by previous research in the field. These aspects include generating referring expressions, deciding determiners, deciding sentence order and preserving rhetorical and anaphoric structure.

5.3.30 Indexing Software For Ancient Kannada Books

S.Settar, Sanjoy Goswami, Abhishek H.K., *Language Engineering Conference, University of Hyderabad, India, Dec. 2002.*

Abstract

This paper deals with the automatic generation of index to Kannada Documents. The basic objective of this exercise is to provide an efficient, user-friendly and reliable tool for index generation of Kannada Documents. The input to the system may come either from an Optical Character Recognition system if it is made available, or from typeset documents. The output provides an editable and searchable index. Results indicate that the application is fast, comprehensive, effective and error free.

5.3.31 Poongkuzhali-Intelligent Tamil Chatterbot

S.Sowmya, *MCA Project, Anna University, Chennai.*

Abstract

A Chatterbot is a tool for Natural Language Communication between and machine. Poongkuzhali is a chatterbot that simulates human conversation through Artificial Intelligence – a program to chat with the system in Tamil.

This software enables a native Tamilian to learn about the internet and other technical topics using the computer, in the form of a lively conversation in his own mother-tongue.

Poongkuzhali takes the user's input, decomposes the input to get the minimal context, reassembles it and returns an appropriate response. The given input is tokenized into words and fed to a Morphological Analyzer that returns the root and identifies the type of question definition, description, pronoun reference, negation, active or passive voice. The simple and compound technical terms are identified and stored separately. The history details are also stored and processed.

A linked list of nodes is created for the identified non-technical terms, consisting of the word and its priority. The list of links for the highest priority word is created. If the input matches any item in this list, the minimal contest is identified. The Knowledge Base is looked up for this context and if it is available, the string is extracted and returned as the answer.. In case the highest priority word fails to provide a match in the knowledge Base, the next highest priority word is considered. This way, the system allows for graceful degradation.

If the input contains pronouns instead of technical terms, the last history item replaces it. When there is no input from the user, a dialogue is initiated by the system. Poongkuzhali is a lively conversationalist, providing an educative entertainment.

5.3.32 A Knowledge Based Approach For Automatic Software Document Generation

G.V. Uma, Dr. T.V. Geetha, *School of Computer Science and Engineering, Anna University. Trends in Software Engineering Process Management (TSEPM) Spring 2002*

Abstract

Natural language generation (NLG) plays a significant role in generating a well-organized text. Extending NLG to generate the necessary documents during software development is a highly challenging task because different classes of users use different types of documents and manuals. Documents have to be produced according to appropriate standards so as to have consistent appearance, structure and quality.

While producing documents, the three phases of natural language generation: content determination,

planning and realization, are used. Content determination uses various inputs from the software requirement specification (SRS) techniques, (Data Flow Diagram (DFD), Entity relationship diagram (ERD) and the state transition diagram (STD) along with various types of knowledge, structural knowledge, syntactic knowledge, domain knowledge, background knowledge, technical knowledge, and linguistic knowledge. It then represents this acquired knowledge using frames. Frames are innately difficult to adapt, so it is enhanced by adding a new component called perspective descriptors.

In planning phase, the identified contents are organized and extracted according to the document standards. Using design patterns based on linguistic concepts it generates phrases for the document generation.

In realization, syntactic and semantic rules are applied to the phrases and the proper document is generated. This paper describes the knowledge-based approach for automatic software document generation.

5.3.33 A Knowledge Based Approach For Automatic Software Document Generation Using Design Patterns

G.V. UMA, *Thesis submitted in May 2002.*

Abstract

This thesis discusses the design and development of knowledge based approach for automatic document generation using design patterns. Natural Language generation is a highly complex task whose automation is performed by accessing huge amount of knowledge sources such as domain knowledge, world knowledge and common sense knowledge. Software documentation is chosen as application for our work because of its closed ended domain i.e. even it the domain is not clearly specified the inputs required have to be properly identified for producing corresponding document. As in any other NLG system, the first task is content determination phase. Inputs such as concepts, entities, relations, structure, sub process, process, states, transitions, events, triggers, time, operations, source, destination, results, and roles are identified. Generally, inputs needed for documenting a project, are obtained from various software requirement specification techniques such as data flow diagrams, entity relationships diagrams, state transition diagrams and control flow diagrams. In the next module, identified inputs for generating a natural language text are organized using knowledge representation technique, frames. In this work, the frame representation scheme has been enhanced with specially designed new component called perspective descriptors, which tries to improve the epistemological status of the frame structure and adapts to changes in the software.

In order to represent the process sequentially among the software entities, casual links have been introduced in the knowledge representation schema in addition to the hierarchical links that depict the structural organization among them. The next phase of software document generation is planning. Adaptive planning is required to deal with producing documents at different levels. For automatic software document generation a generic standard which consists of Aim, Introduction, Purpose, Objective, Functional Behavior, Informational Behavior, Procedural details and all sub process details using document design patterns, has been used. The document design patterns are designed to satisfy the linguistic constraints also. Generally, patterns provide recurrent solutions. For each and every component of the software standard, specific patterns such as initiator, instantiator, illustrator, comparator, counter, fetcher etc. that matches with the linguistic constraints have been designed. The output of the planner is passed to the realization phase to obtain the complete and perfect document.

Realization, maps the identified content from the content determination phase organized using planner phase and generates actual text, which meets syntactic and semantic constraints. The realization tasks are: structure realization and linguistic realization. Structure realization deals with choosing mark-up to convey document structure and linguistic realization deals with insertion of function words, choosing correct inflection of content words, order words within a sentence and applying rules. Based on the three views viz. functional view, behavioral view and informational view, a generative grammar has been designed in order to obtain a proper and perfect software document.

5.3.34 Automatic Software Documentation In Tamil

G.V. Uma, N.S.M. Kanimozhi & T.V. Geetha, *3rd International Conference on South Asian Languages (ICOSAL- 3).*

Abstract

In general, Natural Language Generation is performed in three phases namely, content determination, planning and realization. For automatic software documentation, the various inputs, needed to effectively generate document, have to be identified. These inputs are collected from various software requirement specification techniques such as Data flow diagrams. Entity relationship diagrams and state transition diagrams during content determination phase. Gathered inputs are in the form of entities, actions and corresponding relations between them. In order to provide conceptual and casual interconnection

between the entities, relations and actions we need an effective knowledge representation scheme. A modified frame structure, frames with perspective descriptors have been designed. These perspective descriptors identify the reusable components, which are needed in order to enable effective adaptation in the context of modification to the original software. In this paper specially abstracted design patterns have been designed for document generation. Theses document design patterns are based on linguistic concepts. An abstract plan based on available documentation standards such as NASA, IEEE, ISE, the document design patterns and the domain specific information available in the Knowledge Representation is produced. The abstract plan thus produces ordered document design patterns incorporated with actual application knowledge. The next phase is the realization phase. In this paper an attempt has been made to produce the document in Tamil. For this purpose a bilingual English – Tamil lexicon of the application domain is necessary. A generative Tamil grammar which syntactically reorders the phrases obtained from the abstract plan and attaches appropriate cases to the nouns, is designed.

5.3.35 Automatic Software Documentation Using Frames And Casual Link Representation

G.V. Uma and T.V. Geetha, *School of Computer Science and Engineering, Anna University. Vivek Vol.4, No.3, July 2001.*

Abstract

Generation of natural language text requires the ability to determine what information to include and how to organize it, in order to achieve the communicative goal effectively. Natural language generation is a highly complex task whose automation is performed by accessing various types of knowledge such as Domain knowledge, World knowledge and commonsense knowledge. Software documentation is chosen as the application because even if the format of the inputs is not clearly specified, the inputs are identified properly and proper documentation is generated. Since different levels of documents have to be produced depending on the type of the user, the inputs have to be identified accordingly. VISHAYA NIRNAYEE, a CASE tool has been developed for Content Determination that automatically identifies the inputs from the following software Requirement Specification (SRS) techniques: Data Flow Diagrams (DFD), Entity Relationship Diagrams (ERD) and State Transition Diagrams (STD). The main aim of this work is to provide a casual link representation along with the conceptual link between the various inputs (both functional and non functional requirements), that has already been identified for generating a natural language text in the form of software documents using a frame based knowledge representation (KR). An additional component called Perspective Descriptor is included in the frames in order to provide adaptability, flexibility and reusability. To represent the commonsense knowledge about the domain of software documentation, effort has been made to construct casual links along with the conceptual links using perspective descriptors in frames.

5.3.36 Generation Of Natural Language Text Using Perspective Descriptor In Frames

G.V. Uma and T.V. Geetha, *School of Computer Science and Engineering, Anna University, Chennai. IETE Journal of Research, Vol 47, Nos 1 & 2, January-April 2001, pp 43-56*

Abstract

Generation of natural language text requires the ability to determine what information to include and how to organize it to achieve its communicative goal effectively. Natural Language generation is a highly complex task whose automation is performed by accessing huge amount of knowledge sources such as domain knowledge, world knowledge and common sense knowledge. Software documentation is chosen as application for our work because of its closed ended domain i.e. even if the domain is not clearly specified the inputs required have to be properly identified for producing corresponding document. Since different levels of documents (Documenting each one of the them is a critical job. The documentation is nothing but the art of accumulation, classification and dissemination of information) have to be produced depending upon the status of the user, the inputs have to be identified accordingly. The main aim of this work is to identify necessary inputs needed for generating a natural language text in the form of software document using knowledge representation technique, frames, by adding a new component called perspective descriptors, by which the epistemological status of the frame is being improved which increases the efficiency of the search engine and tries to reduce the spatio temporal complexities obtained in the existing frame structure. Frames are chosen as knowledge representation scheme because of their efficiency in processing and producing huge amount of information without any deterioration.

5.3.37 Soft Plan – A Planner For Automatic Software Documentation

G.V. Uma and Dr. T.V. Geetha, *National Conference on Document Analysis and Recognition, Mandya, Karnataka – 13th and 14th July 2001.*

Abstract

Text planning in Natural language generation plays a significant role in generating well-organized text. Natural Language generation is performed in three namely content determination, planning and realization. Content determination deals with identifying inputs, planning organize the identified inputs based on standards and templates. Realization maps content determination and planning and produces the text satisfying syntactic and semantic constraints. As a first step towards automatic software documentation, the various inputs needed to effectively generate all types of manuals have to be identified. In the content determination phase, the inputs are collected from various software requirement specification techniques such as Data Flow Diagram, Entity relationship diagram and state transition diagrams. Gathered inputs are in the form of entities, actions and corresponding relations between them. In order to provide interconnection between the individual entities, relations and actions we need an effective knowledge representation scheme. In this work the existing frame structure is modified using perspective descriptors for enhancing adaptability for automatic generation of software documents. The planning for the text generation differs form the general AI planning. While general AI planning deals with states and state dictated application of operators, in text planning, the application of operators is dictated by content organizational constraints. Software documentation deals with close-ended domains. Adaptive planning is required to deal with producing documents at different levels. Hence we have designed document design patterns that take care of matching with the software standards. Software standards are very important in software documentation because it provides consistent appearance and structure. Software documentation has to be done at various levels and also base on the status of the user. Different types of manuals have to be prepared for different set of users to meet their own requirements.

5.3.38 Malayalam Spell Checker

Santhosh. T. Varghese, K.G. Sulochana, R. Ravindra Kumar, *RCILTS-Malayalam, International Conference on Universal Knowledge and Language, Goa, November, 2002.*

Abstract

This paper discusses the various stages involved in the development of a Malayalam Spell Checker, which is a tool that will check the spelling of the words in a Malayalam text file, validate them and in case of error, list out the right spelling in the form of suggestions. The structure of Malayalam language is such that a large number of words can be derived from a root word and also any number of words can be concatenated with a root word based on their use. Hence a purely dictionary based approach for Spell Checking is not practical. A 'Rule cum Dictionary' based approach is followed in the Design of the Spell Checker. The grammatical behavior of the language, the formation of words with multiple suffixes, concatenation of multiple words with suffix/suffixes in between and the preparation of the language module are dealt with in detail.

The different modules in the Spell Checker Engine are: Morphological Analyzer, Morphological Generator and Error detection and Suggestion Generator. The Morphological Generator splits the input word into valid suffixes and root words and associates a paradigm number to distinguish the Part Of Speech (POS) to which it belongs. The Morphological Generator is to crosscheck the POS identified by the Morphological Analyzer with the help of a set of grammatical rules known as Sandhi (junction) rules. The Suggestion Generator module generates suggestions to aid correction of a misspelled word. The root word dictionary in the Spell Checker contains about 9500 words, which includes some words taken from other languages also. It contains two lists, one for parts of speech and the other for postpositions. The Spell Checker Engine is capable of handling minor grammatical errors like concatenation of wrong suffixes. The beta version of the Spell Checker is tested with 400 pages of data and average error detection rate is around 90%.

5.3.39 Computer Processing Tools For Indian Languages

Om Vikas, *Electronics Information & Planning, May 1991*

Abstract

This paper presents the scenario of technology development for Indian languages including the aspects of standardisation of codes for information interchange and keyboard layout, input-output devices, potential application areas such as machine translation, learning systems, speech input-output. Lot of interest has got generated during the present decade. Now, the programming, query and publishing environments are becoming available in Indian languages. Availability and economics of proven software packages are the determining factors in popularizing computer processing in non-English languages. Common standardisation of phonetic based world languages and international cooperation for multi-lingual software development will be necessary. However, there is need for extensive promotional efforts to take this emerging technology to the people at large.