Week 2&3 (30/09/13 – 12/10/13)

1. **Literature Survey**

The following reference papers were analysed during the literature survey phase.

[a]. A Copy detection Method for Malayalam Text Documents using N-grams Model - Sindhu.L, Bindu Baby Thomas, Sumam Mary Idicula

In this paper a method of copy detection in short Malayalam text passages is proposed. Given two passages one as the source text and another as the copied text it is determined whether the second passage is plagiarized version of the source text. An algorithm for plagiarism detection using the n-gram model for word retrieval is developed and found tri-grams as the best model for comparing the Malayalam text. Based on the probability and the resemblance measures calculated from the n-gram comparison , the text is categorized on a threshold. Texts are compared by variable length n-gram(n={2,3,4}) comparisons. The experiments show that trigram model gives the average acceptable performance with affordable cost in terms of complexity.

[b]. An Unsupervised Approach To Develop Stemmer - Mohd. Shahid Husain

This paper presents an unsupervised approach for the development of a stemmer (For the case of Urdu & Marathi language). To train the system training dataset, taken from CRULP and Marathi corpus are used. For generating suffix rules two different approaches, namely, frequency based stripping and length based stripping have been proposed. The evaluation has been made on 1200 words extracted from the Emille corpus. The experiment results shows that in the case of Urdu language the frequency based suffix generation approach gives the maximum accuracy of 85.36% whereas Length based suffix stripping algorithm gives maximum accuracy of 79.76%. In the case of Marathi language the systems gives 63.5% accuracy in the case of frequency based stripping and achieves maximum accuracy of 82.5% in the case of length based suffix stripping algorithm.

[c]. A Literature Review: Stemming Algorithms for Indian Languages - M.Thangarasu, Dr.R.Manavalan

This expository paper presents survey of some of the latest developments on stemming algorithms in data mining and also presents with some of the solutions for various Indian language stemming algorithms along with the results. Some of the methods analysed in this paper are Morphological Analyzer, Porter Stemming Algorithm, Assas-Band, Brute Force Algorithm, Finite State Automata, Suffix Stripping etc.

[d]. Stemmers for Tamil Language: Performance Analysis - M.Thangarasu, Dr.R.Manavalan

Tamil Language raises several challenges to NLP, since it has rich morphological patterns than other languages. The rule based approach light-stemmer is proposed in this paper, to find stem word for given inflection Tamil word. The performance of proposed approach is compared to a rule based suffix removal stemmer based on correctly and incorrectly predicted. The experimental result clearly show that the proposed approach light stemmer for Tamil language perform better than suffix removal stemmer and also more effective in Information Retrieval System (IRS).

[e]. Hybrid Inflectional Stemmer and Rule-based Derivational Stemmer for Gujarati - Kartik Suba, Dipti Jiandani and Pushpak Bhattacharyya

In this paper they present two stemmers for Gujarati- a lightweight inflectional stemmer based on a hybrid approach and a heavytheyight derivational stemmer based on a rule-based approach. Besides using a module for unsupervised learning of stems and suffixes for lighttheyight stemming, they have also included a module performing POS (Part Of Speech) based stemming and a module using a set of substitution rules, in order to improve the quality of these stems and suffixes. The inclusion of these modules boosted the accuracy of the inflectional stemmer by 9.6% and 12.7% respectively, helping them achieve an accuracy of 90.7%. The maximum index compression obtained for the inflectional stemmer is about 95%. On the other hand, the derivational stemmer is completely rule-based, for which, they attained an accuracy of 70.7% with the help of suffix-stripping, substitution and orthographic rules. Both these systems theyre developed to be useful in applications such as Information Retrieval, corpus compression, dictionary search and as pre-processing modules in other NLP problems such as WSD.

[f]. A Lightweight Stemmer for Hindi - Ananthakrishnan Ramanathan, Durgesh D Rao

This paper presents a lightweight stem- mer for Hindi, which conflates terms by suffix removal. The proposed stem- mer is both computationally inexpensive and domain independent. The paper dis- cusses the systematic manner in which the suffix list was developed, and pro- vides the linguistic rationale behind in- cluding various suffixes in the list. Simi- lar techniques can be used to build stem- mers for other Indian languages such as Marathi, Gujarati, and Punjabi. The stemmer has been evaluated by comput- ing understemming and overstemming figures for a corpus of documents. The results are favourable and indicate that the proposed stemmer can be used effec- tively in IR systems.

[g]. A Simple Stemmer for Inflectional Languages - Jiaul H. Paik , Swapan K. Parui

In this paper they present an unsupervised stemmer which is based on a simple corpus and is for several Indian Languages. The algorithm is tested for three major Indian languages – Bengali, Hindi and Marathi, which are primarily suffixing in nature. It implemented blind and corpus specific stemming method. The paper also describes about test collections, weight assignment to words, and query document matching technique used.

[h]. YASS: Yet Another Suffix Stripper - Prasenjit Majumder, Mandar Mitra, Swapan K. Parui, And Gobinda Kole

In this article, they describe a clustering-based approach to discover equivalence classes of root words and their morphological variants. A set of string distance measures are defined, and the lexicon for a given text collection is clustered using the distance measures to identify these equiv- alence classes. The proposed approach is compared with Porter's and Lovin's stemmers on the AP and WSJ subcollections of the Tipster dataset using 200 queries. Its performance is comparable to that of Porter's and Lovin's stemmers, both in terms of average precision and the total num- ber of relevant documents retrieved. The proposed stemming algorithm also provides consistent improvements in retrieval performance for French and Bengali, which are currently resource-poor.

2. Problem Definition

The availability of digital information has made it possible to use digital data which is also the cause of the misuse of the available data Some of the issues associated with the misuse of digital data are Plagiarism detection, ownership identification etc. Plagiarism detection is of particular interest to people in the academia and the publishing sector. Plagiarism means copying thought and text of another person and presenting them as ones's own work. English copy detection systems have been studied since 1990s and some of them can be freely downloaded from the Internet while a copy detection system for Malayalam is not available.

Week 4 (14/10/13 – 19/10/13)

1. Framework Identification

A framework called SILPA (Swathanthra Indian Language Processing Applications) exists for the field of Natural Language Processing in Indic Languages. This framework is selected as the base framework for project due to the availability of various language manipulation utilities provided in the framework.

2. Language Selection

Python is selected as the programming language with PyGTK for designing GUI because of the availability of strong string manipulation and regular expression functions in Python. Python also provides Unicode support which is essential for Indic Languages as they are not coded in ASCII.

Week 5 (21/10/13 – 26/10/13)

1. SILPA (Swathanthra Indian Lanugage Processing Applications)

Silpa, Swathanthra Indian Language Processing Applications is a web platform to host the free(dom) software language processing applications easily. It is a web framework and a set of applications for processing Indian Languages in many ways. Or in other words, it is a platform for porting existing and upcoming language processing applications to the web. Silpa can also be used as a python library or as a webservice from other applications.
As it is meant for covering all languages of India, all modules should be capable of handling all scripts from India(Sometimes English too). At the same time , the language of input data is transparent , meaning, user need not mention that _this_ is the language in which he/she is entering the data. Unlike desktop applications which asks to specify the language along with the input data(for eg: Spell checker) , the modules should try to detect the language them self. And if possible, modules try to process the data even if the input data is in multiple Indic scripts. The modules may be General purpose(eg: Dictionary, Spellcheck,Sort. Transliteration, Font conversion..) or Technology/Algorithm Demonstration purpose (eg: Hyphenation, Stemmer, Search algorithms)

Week 6 (28/10/13 – 2/11/13)

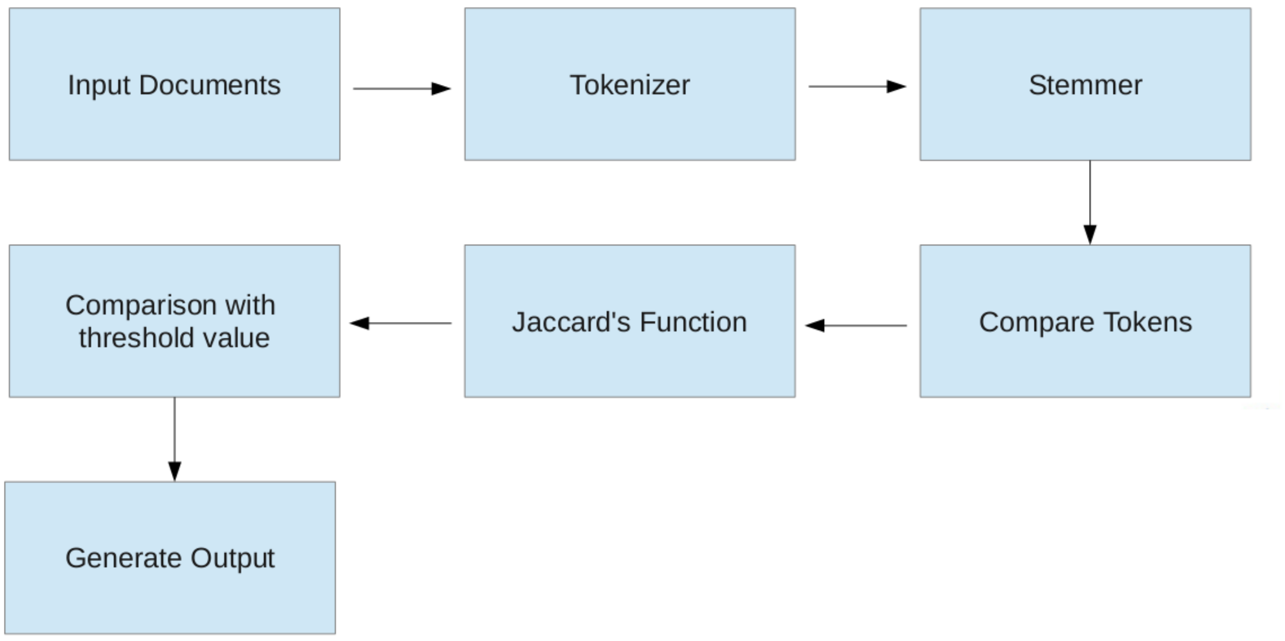
1. Deciding Solution

Inorder to solve the special problem of Inflection which is applicable commonly for Dravidian languages, the method of stemming was decided to be used. It involves finding the root word of each important token in a document. Then these root words are collected and compared to each other for calculating the similarity.

The GUI will consist of text boxes to input the two documents and a result area where the percentage similarity of two documents will be shown.

Week 7 (4/11/13 – 9/11/13)

1. Deciding overall structure



Tokenizer – It splits the document into sentences and each sentence into tokens(words). Common words, called stop words, may be removed to reduce the time complexity of the algorithm.

Stemmer - It finds out the root word of each token in the document based on rules defined previously. The root words are stored in a dictionary to be imported in the program.

Compare Tokens – In this step, the tokens of each sentence are arranged in ascending order and compared for equivalency.

Jaccard's Function – This function is used to calculate the similarity of each pair of token set in both documents.

Threshold Value – A user (or programmer) defined threshold value is used to decide whether the documents are near duplicates. However, the percentage of similarity is displayed irrespective of this threshold value