IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-11, Nov 2018 E-ISSN: 2347-2693

System Implementation and Testing of Proposed Language Independent Stemmer

M. Kasthuri[1*]

Dept. of Computer Science, Bishop Heber College (Autonomous), Tiruchirappalli – 620 017, Tamil Nadu, India

[*]*Corresponding Author: stephenbasilkasthuri@gmail.com*

Available online at: www.ijcseonline.org

Accepted: 13/Nov/2018, Published: 30/Nov/2018

***Abstract*—** Information Retrieval (IR) is an emerging discipline that involves methods, models and patterns to find the documents of an unstructured nature in the dynamic environment. Information Retrieval System (IRS) retrieves the user required information over billions of documents stored in millions of computers. Search Engine is playing a major role in Information Retrieval Systems (IRS) to identify the morphological variants of the language using Stemming. Stemming is an important pre-processing step in query-based systems such as IRS, Web Search Engine, Natural Language Processing (NLP), Big Data Analysis, etc. The purpose of stemming is to diminish different grammatical or word forms to a common base form. In this digital era, most of the web pages are designed using English and European languages. Similarly, the web pages designed with Indian and other Asian languages are also increasing. Search Engines available in Information Retrieval Systems required for dealing with the morphologically different languages in every fraction of a second. The study reveals that the approaches for developing the stemmer involve rule-based, machine learning and hybrid approach. However, each one of them has its own limitations. Therefore, it has been proposed to design the model for Language Independent Stemmer using Dynamic Programming (DP) to retrieve the multi-linguistic web documents with the greater speed and accuracy. However, this research paper presents system implementation and testing of Proposed Language Independent Stemmer (PLIS). The performance of the proposed LIS has been analyzed using a test bed.

***Keywords*—**Information Retrieval, Stemming, EMILLE, Language Independent Stemmer, Dynamic Programming.

I. INTRODUCTION

In recent years, IR has been established as one of the research disciplines in Computer Science with growing industrial impact. With the growth of the World Wide Web, Information and Communication Technologies, and high-speed Internet connections, the generation and transmission of large volume of data across the world have increased over the last decade. The networks, technologies and information which are being generated require faster and better Information Retrieval Systems. Due to the increase of information day by day, the search engines require more efficient techniques for retrieving the data faster and with great accuracy. Web document in a large number of Indian languages like Hindi, Urdu, Bengali, Oriya, Tamil, Telugu and Marathi is now available in the electronic form [1, 2, 3, 4]. Information Retrieval System (IRS) plays a vital role in providing access to this information. Stemming is the important pre-processing step in Information retrieval, and it finds the root or stem from the given inflected word. In search engine terminology, stemming is the comparison of the input query to the root form of a word and identifies its morphological variants available in the web documents [5]. For example, a user may search for the term cheaper, but a search engine may return the search results with the root form of the given word such as cheap, cheapen, cheaper. This makes stemming an attractive option to increase the ability of matching the query and document vocabulary in the Search Engine. Word stemming is an important feature supported by present-day indexing and search systems. Indexing and searching are part of Text Mining applications, Natural Language Processing and Information Retrieval Systems. The Proposed Framework for Language Independent Stemmer [6], PLIS algorithm and Illustrative examples presented in the previous research paper [7]. The performance of the PLIS has been evaluated using EMILLE corpus. Proposed Architecture of Language Independent Stemmer also exists in the previous research paper [8]. This research paper presents only the System Implementation and Testing of Proposed Language Independent Stemmer.

The Section I contain the introduction of information retrieval, stemming approaches available for various languages. Section II contains the related work of stemming approaches, Section III contains Proposed Implementation for Language Independent Stemmer, Section IV concludes research work of proposed language independent stemmer.

II. RELATED WORK

Husain, (2012) has proposed an unsupervised language independent stemmer for Urdu and Marathi languages. This

stemmer hybridized with two existing stemming approaches such as rule-based and n-gram approaches [1]. The main advantage of this stemmer is that it is a language independent stemmer. However, the important problem in this stemmer is that under-stemming and over-stemming errors can occur.

Saharia et al. (2013) have proposed a stemmer for Assamese language using Hidden Markov Model and rule-based. The experimental study has been done on the bases of EMILLE corpus, and the results showed that single letter suffixes have provided more accuracy than multiple letter suffixes [9, 17]. However, it is a language dependent stemmer. The rule based approach requires an exhaustive linguistic knowledge to find the stem words and this limits the wide usage of this stemmer.

Joshi et al. (2014) have proposed a stemmer for Punjabi language using synset approach. This stemmer uses table lookup and rule-based approach. Synset approach finds the stem words by using synonyms of the given word. This stemmer attempts to provide better results using different algorithms in a hybrid way [10]. However, the main issue identified in this stemmer is limited Punjabi words only are stored in the table, which makes to produce incorrect stem word.

Nehar et al. (2015) have proposed a stemmer for Arabic language using Trigram, Transducers and Rational Kernels approach [14]. The experimental result shows that stemming improves the quality of classifiers in terms of accuracy, recall and F1-measure. However, the main limitation of this stemmer is that trigram approach requires large storage space and it is not a practical approach. The rational kernels classification is computationally expensive.

Anusha et al (2016) have proposed Hindi Stemmer for Information Retrieval System. This proposed stemming algorithm was implemented for Hindi Noun words [11]. This Stemming algorithm gives accuracy of 92.2%. However, this stemmer does not support multi linguistic web document retrieval.

Adege et al. (2017) have generated a stemmer of Ge'ez language using rule-based approaches [12]. The experimental result shows that, this research work performed with an accuracy of 82.42%. However, limited rules sets were created, which affects the accuracy of the proposed stemmer. It requires linguistic knowledge to generate rule set.

Robert et al. (2018) have presented Experimental Analysis of Stemming on Jurisprudential Documents Retrieval. It is less aggressive stemmers, presented the best cost-benefit ratio, since they reduced the dimensionality of the data and increased the effectiveness of the information retrieval evaluation metrics in one of the analyzed collections [13]. However, this stemming research work mainly concentrated on jurisprudential documents.

III. PROPOSED IMPLEMENTATION FOR LANGUAGE INDEPENDENT STEMMER

3.1.Experimental Setup

The framework for the Proposed Language Independent Stemmer is simulated and tested by establishing a Test Bed. The development phase starts with the requirement input queries delivered by the requirement phase and maps the requirements into architecture. The User Interface (UI) for the input query as well as Application Server (AS) for stemming are developed. Then the test bed is also used to test the performance of the proposed system. During the second phase, the supporting tools are built for query processor and Search Engine. After completing this development phase, the user and server applications with required software tools are integrated in the test bed. Figure 1 shows the test bed environment.

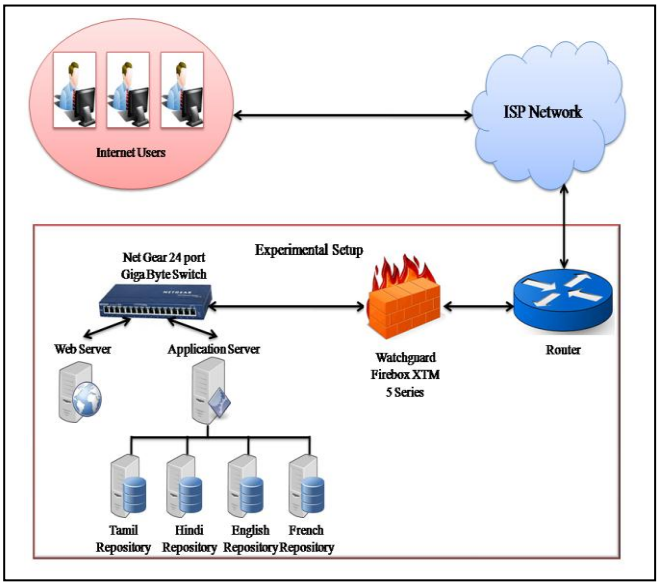

Figure 1. Test Bed Environment

Choosing the right structure for an application is critically important. Bad architectural choices cannot be usually fixed during implementation, no matter how good the developers are. Making the wrong decisions leads to lower performance, less accuracy, etc. Hence, the PLIS architecture has been effectively deployed on an experimental test bed to implement the proposed system and performance, accuracy and strength of the PLIS are measured.

3.2. Hardware Setup

The Hardware setup of the PLIS system is divided into seven entities. They are Internet enabled User System (US), Web

Server (WS), Application Server (AS), and Repository Servers (RS) for English, French, Hindi and Tamil. The Hardware setup of the User System (US) is configured with Intel Core™ i3 processor CPU 3.06 GHz dual core with 2 GB RAM and running Windows 7 operating system, Apache Version 2.2.8 and PHP Version 5.2.6. The user device is incorporated with User Interface application. The user gives the input query using this UI. The Hardware setup of the Web Server (WS) is configured with Intel Xeon 5600 processor, 32 GB RAM, 300 GB x 2 |600 GB x 3HDD and running on Windows 2008 operating system. After stemming, the extracted Web URLs are maintained in the Web Server. The Hardware setup of the Application Server (AS) is configured with Intel Core i5 Processor, 8GB DDR3 RAM, 500GB SATA HDD, and running on Windows 2003 operating system Apache Version 2.2.8 and PHP Version 5.2.6. The application server contains query processor and Search Engine applications. The Hardware setup of the English Repository Server (ERS) is configured with AMD C-60 with Radeon (tm) Processor, 2GB RAM and running on Windows 7 operating system. The English repository device contains collections of English words, which are collected from different web sites. The Hardware setup of the French Repository Server (FRS) is configured with Intel Xeon 3 GHz dual processor with 4 GB RAM, running the Linux Fedora Core 7 operating system.

The French repository device contains collections of French words, which are extracted from web sites. The Hardware setup of the Hind Repository Server (HRS) is configured with Intel Core™ i3 processor CPU 3.06 GHz dual core with 4 GB RAM and running Windows 7 operating system. The Hindi repository device consists of collections of Hindi words, which are extracted from EMILLE corpus [EMILLE, 2014].The Hardware setup of the Tamil Repository Server (TRS) is configured with Intel Core™ i5 processor CPU 3.06 GHz dual core with 4 GB RAM and running Windows 7 operating system. The Tamil repository device contains collections of Tamil words, which are collected from EMILLE corpus. To protect the experimental network from the intruders, the firewall is configured with Watchguard XTM 5 Series and the switch is configured with Netgear FS728TP ProSafe 24-port 10/100 Smart Switch w/4 Gigabit.

3.3. Software Setup

The test bed is implemented based on PLIS framework, where the functional components are implemented using PHP 5.4, Html5, JavaScript and CSS3. Generally, PHP is a server-side scripting language designed for web development but PHP is also used as a general purpose programming language. PHP was mounted on more than 240 million websites and 2.1 million web servers in January 2016. Therefore, PHP is the preferred environment to deploy the PLIS and other supporting functionalities such as user interface, query processor and search engine. The main advantages of JavaScript are simple, versatile and effective language that used to extend functionalities in websites. It is used to shape the behavior of web pages. Cascading Style Sheets (CSS) is used to define the look and layout of the text and other materials involved in PLIS and its supporting applications. Using CSS3, the audio, video and all types of web pages are tagged and viewed without the support of third party plug-ins. Moreover, the World Wide Web Consortium (W3C) encourages the use of CSS over explicit presentational HTML. The majority of the websites are created as visually engaging web pages using HTML, JavaScript and CSS. Therefore, the functional components available in the proposed framework are designed using HTML5, JavaScript, CSS3 and PHP 5.4.

3.4.System Implementation and Testing

The information retrieval system is developed using PHP 5.4, HTML5, JavaScript and CSS3, which is the suitable environment to test the proposed language independent stemmer. Internal structure of the proposed system consists of three modules:

1. User Interface Module – handles creation of all user interfaces, related objects such as forms, text box, button and scroll bars and etc.

2. Query Processor Module – deals with natural language input query, remove stop words, find stem words using PLIS and query matching after retrieving web links form search engine.

3. Search Engine Module – used to crawl the web sites from Internet, construct the web pages repository, extract the text from web pages, remove stop words and generates stem words using PLIS.

3.4.1. Implementation of User Interface

The User Interface (UI) for the proposed system is a basic requirement to obtain the user input query, generating the stem words and to establishing the communication with the Web Server. The user interface was implemented using PHP. The main advantages of PHP are, used for creating server side as well as client side scripts over competing technologies and easier implementation. Figure 2 shows the user interface with input query and stop words are removed using PLIS. Hence, the Proposed Language Independent Stemmer is the important pre-processing approach. The proposed system needs to extract relevant web links based on the user input query after proceeds the stemming. Recently, most of the web sites are mounted with PHP and Apache Web Server. The server performs various operations such as Web Crawling, Text Extraction, Sentence and Words Split, Stop Words Truncation and Stemmer Generation. The above

functionalities are split and assigned to various servers such as Web Server (WS), Application Server (AS) and Repository Servers (RS). The user and server communication are established using Apache. The main advantage of Apache is open source software and its source code can be modified to suit individual needs. This gives Apache a significant advantage over almost all of its competitors without sacrificing any features.

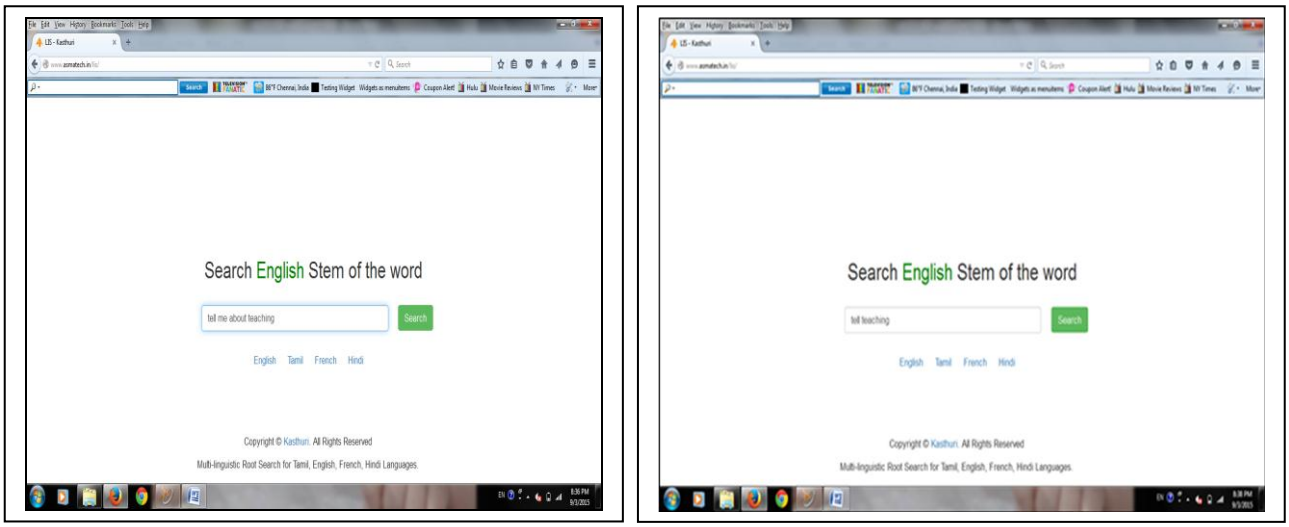

Figure 2. User Interface with Input Query

The open source status also eliminates the recurring license and support fees required to continue running other web server programs. The WS is useful for maintaining web URLs extracted from the Internet. The AS maintains the supporting applications of the proposed system such as Query Processor (QP) and Search Engine (SE). The AS performs the stemming and generating the relevant web links, which is sent to the User Interface (UI). There are four Repository Servers are available for storing words such as English Repository Server (ERS), French Repository Server (FRS), Tamil Repository Server (TRS) and Hindi Repository Server (HRS). Hence, all the functional components available in the proposed servers were designed using PHP.

3.4.2. Implementation of Query Processor

The User Interface (UI) sends the user input query to the Query Processor (QP). The QP consists of three sub modules:

1. Stop Words Remover (SWR) – It is used to remove all stop words from the input queries.
2. PLIS – It generates stem words for the input queries.
3. Query Matcher (QM) – It is responsible to match the relevant web sites based on the input stem word.

3.4.2.1. Stop Words Remover

The input query is obtained through the User Interface, and it consists of some commonly used words or less significant words. The Query Processor does not consider these common words in order to reduce the memory and to increase the search results. This module considers each term in the input query and checks with the Stop Words List (SWL). If any words match with the Stop Words List then it will be removed and then remaining words are forwarded to the PLIS module.

3.4.2.2. Proposed Language Independent Stemmer

The PLIS is used to find the stem word for any language using Dynamic Programming. The internal structure of the PLIS consists of five modules:

1. Similar Words Constructor (SWC) Module – Similar words are extracted from local repositories based on the writing morphology of the given language.
2. Character Analyzer (CA) Module – It finds the ED and LCS values for the words sent by the SWC.
3. Rule Based Filter (RBF) Module – It truncates some irrelevant strings which do not satisfy the specified two rules in the proposed algorithm.
4. Words Filter (WF) Module – It is responsible to find the words who reach ED (MAX) and LCS (MIN).

5. Stemmer Generator (SG) Module – It generates the stem word for the given input query using Dynamic Programming.

Figure 3 shows the outcome of the stop words remover and retrieved relevant document.

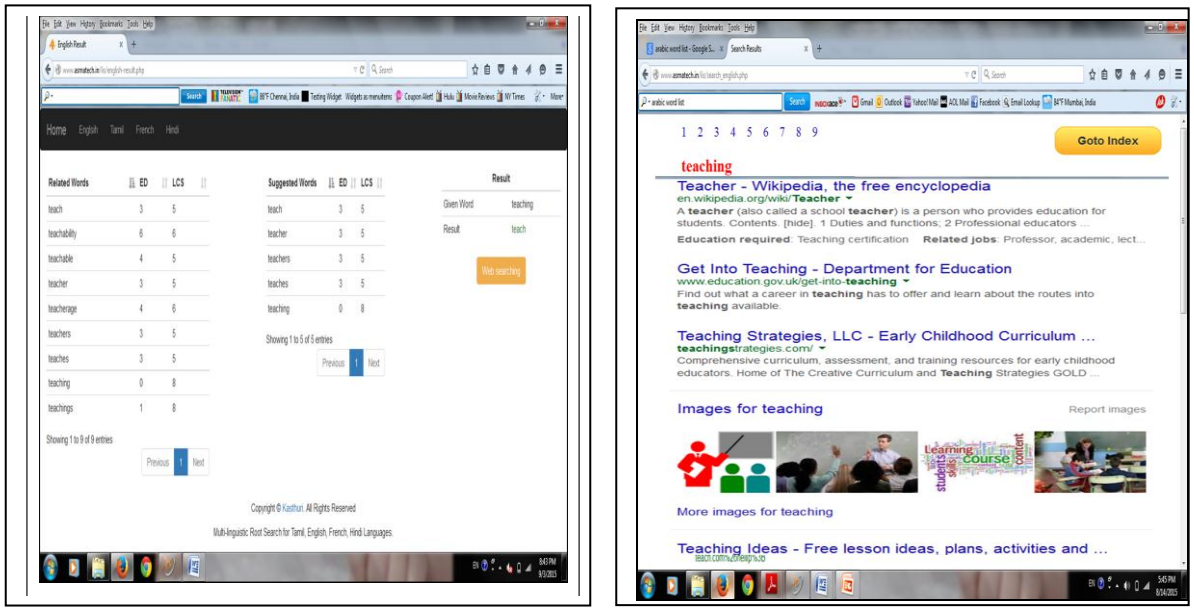

Figure 3. Outcomes of the Stop Word Remover

The SWC module is used to group similar words based on each input terms. It checks the writing style of the given language. If the writing style of the language starts from left hand side, then it collects strings from local repositories and filters these strings whose first three characters as same as the given input terms. If the writing style of the language starts from right hand side, then it finds the reverse of the input string as well as retrieved strings, and then filters the strings whose first three characters are the same as the given input string. After filtration, the words are sent to Character Analyzer (CA) module. It is used to calculate Edit Distance (ED) and Longest Common Subsequence (LCS) for each filtered words along with the given input terms simultaneously. ED and LCS are two important Dynamic Programming concepts. They support for providing the accurate results in less processing time. In order to reduce the irrelevant words, the rule-based filtration process is applied and the words, which do not satisfy the rules mentioned in the proposed algorithm, are truncated.

After applying the rule-based filtration, the proposed algorithm obtains EDs and LCSs for each similar word. The Words Filter module is responsible for selecting the relevant words produced by the RBF. In order to obtain them, WF finds the words that reached ED (MAX) and LCS (MIN). Finally, the relevant words are sent to Stemmer Generator (SG) module. If the WF module sends a single word, then SG considers this word a stem word. Otherwise, if there are more words, then find out the length of the each remaining words and choose the word as stem word whose length is the minimum. If there are more words having the same minimum of length, then compare the selected words with the given input string character-wise.

Finally, after comparison, the minimum length word is considered a stem word, which is having more similar characters compared to the given string and makes it as a stem word.

3.4.2.3. Query Matcher

Once the stem word is generated successfully, the Query Matcher (QM) module available in the Query Processor sends the relevant web sites to the user based on their input queries. It retrieves the relevant web sites with the help of Search Engine procedures. After that the Query Matcher module is responsible to filter the irrelevant web sites and sends only useful relevant web sites to the user.

3.4.3. Implementation of Search Engine

Query Processor passes only the stemmed words of the input query to Search Engine. This module in the proposed system consists of three sub modules:

1. Web Crawler

2. Text Extractor
3. Sentence and Words Splitter
4. Stop Words Remover
5. Stemmer Generator
6. Search Engine Repository Constructor

3.4.3.1. Web Crawler (WC)

The Web Crawler module is systematically browses the World Wide Web (WWW). It is acting like a Meta search engine that blends the top search results from Google search and Yahoo! search. For parsing the web page of a URL, the proposed system uses simple HTML DOM class namely simple_html_dom.php, which is downloaded from SourceForge. This PHP file contains web crawler functionalities and that converts relative URL's to absolute URL's. Then this module intercepts all the URL's of the web pages using web crawler, by which its related sub links are also obtained.

3.4.3.2. Text Extractor (TE)

Text Extractor (TE) is responsible for extracting the useful and meaningful text from the web pages. To achieve this, it removes the html tags, metadata and semantic entities from web pages.

3.4.3.3. Sentence and Words Splitter (SWS)

Sentence and Words Splitter (SWS) is used to generate the sentences initially, by which words are segmented. Collections of words are sent to Stop Words Remover.

3.4.3.4. Stop Words Remover (SWR)

Stop Words Remover (SWR) is used to remove all the stop words. The SWS sends the collection of words to Stop Words Remover. The SWR is responsible to check each word against the Stop Words List (SWL). If the word matches with SWL, then it is removed, otherwise the word is sent to Stemmer Generator (SG) module. The SWR functionality is deployed in Query Processor (QP) as well as Search Engine (SE) modules. In QP the SWR removes the common and less significant words from the input query, where as in SE, it removes the stop words from the web documents.

3.4.3.5. Stemmer Generator (SG)

Stemmer Generator (SG) is responsible for generating stem words for each term extracted from the previous module. It removes duplicate stem words, and the remaining stem words are inserted into stem words list. This module is also employed in both Query Processor and Search Engine. In QP the SG finds the stem word of the input query using PLIS, where as in SE, it finds the stem word of the web documents. Using these modules, the words are inserted into four repositories such as English, French, Tamil and Hindi repositories, which are located in four different devices. The following figures from 4 to 6 depict the system implementation of PLIS on various languages.

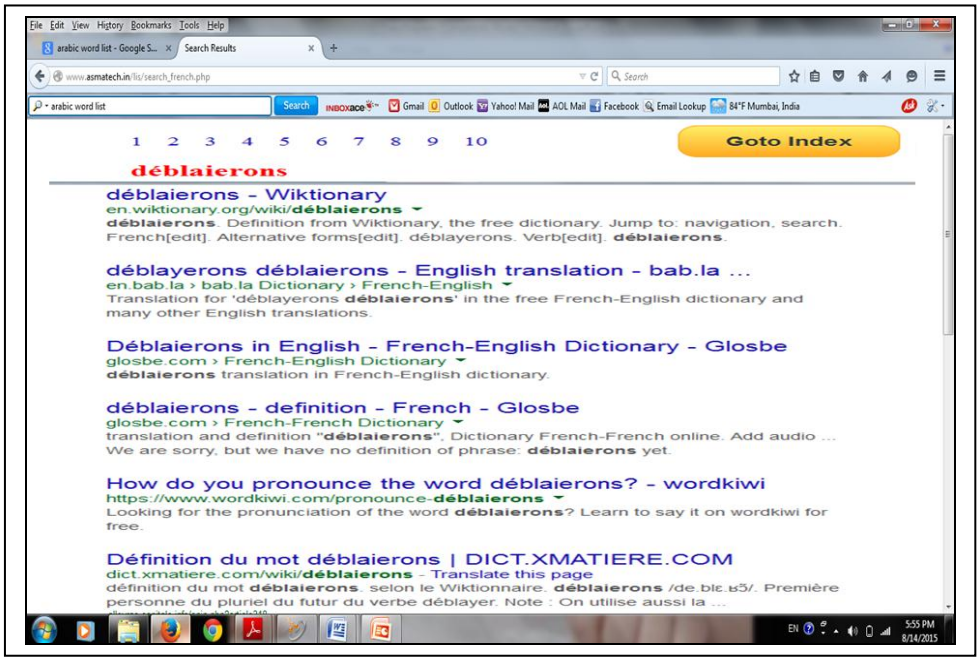

Figure 4. PLIS for French Language

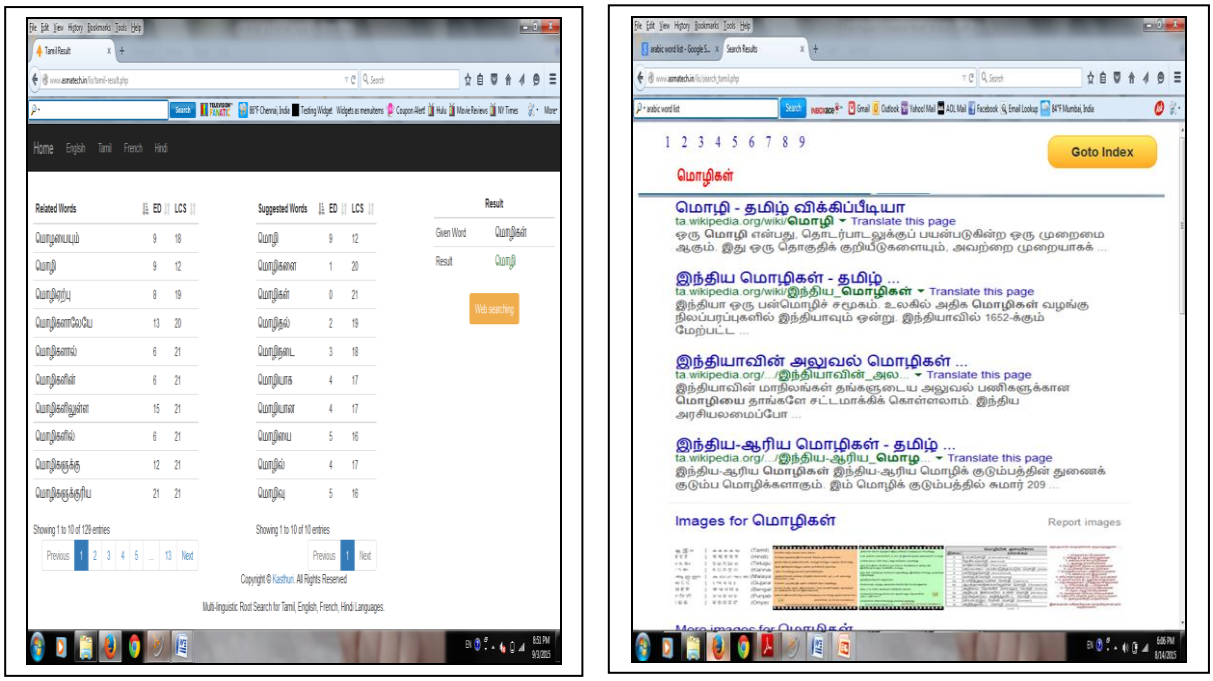

Figure 5. PLIS for Tamil Language and List of Related Web Sites for the Input Query மொழிகள்

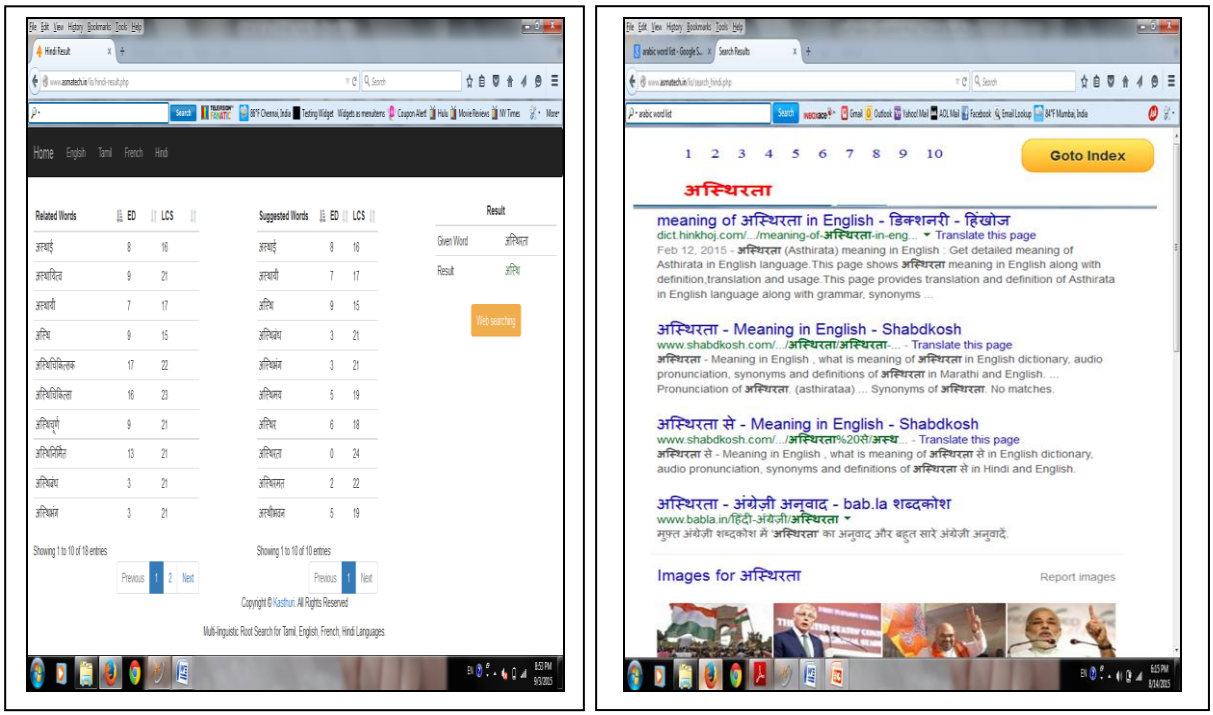

Figure 6. PLIS for Hindi Language and List of Related Web Sites for the Input String अस्थिरता

3.4.3.6. Search Engine Repository Constructor

Once the stem words are generated, the Search Engine module finds the relevant web URLs matched to newly found stem words via Internet. Since the proposed system does not have its own search engine repository, it uses the HTML DOM class namely simple_html_dom.php file to extract relevant web sites using Google search engine repository. The retrieved web sites are sent to the Query Processor once again, where Query Matcher module is responsible for matching the web sites with the input queries. Query Matcher sends the web sites based on input query terms and then sends the web sites based on its morphological variants to the user interface. Finally, retrieved relevant web sites are displayed at the web user.

3.5. Repository Servers

Repository Servers (RS) contains collections of words. There are four Repository Servers available such as English, French, Tamil and Hindi. The strength, accuracy and efficiency of the PLIS are evaluated using test bed, which requires words to find the stem words [15]. English and French repositories are constructed using collections of words extracted from different websites. Collection of words are extracted from EMILLE corpus and inserted into Tamil and Hindi repositories.

Adoption of rule-based, lookup and hybrid approaches are useful and have advantages, there are several limitations and problems [16]. In order to overcome the existing problem, the proposed system has been implemented. The proposed system has tested on various languages like English, French, Tamil and Hindi. The Proposed Language Independent Stemmer generates the accurate result even with agglutinative language like Tamil. The PLIS algorithm considers diacritics and accented characters of the language and generates efficient result. The system implementation incorporates Proposed Language Independent Stemmer (PLIS) both in query side as well as retrieved web document side and PLIS reaches 98.397% of accuracy. Therefore, the PLIS enhances to find the stem words for Indian and Non-Indian languages.

IV. CONCLUSION

Stemming approaches in Information Retrieval Systems focus on increasing the retrieval performance, consuming less time but providing greater accuracy, strength and supporting multi-linguistic documents need more attention [15]. In view of the above aspects, this research work proposes the Language Independent Stemmer for Information Retrieval Systems using Dynamic Programming concepts. However, this research paper presented System implementation of PLIS. The functional components available in the proposed framework are designed using HTML5, JavaScript, CSS3 and PHP 5.4 [5, 6]. The implemented system has ability to generate the stem word in any language automatically without any human interference. Also, the PLIS can be very well adopted in the applications that employ the stemming process. This proposed system can be extended to any Information Retrieval Systems, Search Engine, Natural Language Processing, Machine Translation, Computational Linguistic, etc. Hence, the PLIS is a novel one when compared to the existing stemmers designed for Information Retrieval Systems. Any Information Retrieval System incorporates the PLIS both in query side as well as retrieved web document side to enhance the retrieval performance and accuracy.

ACKNOWLEDGEMENT

I wholeheartedly thank University Grants Commission (UGC), Hyderabad for sanctioning the grant to complete the Minor Research Project entitled Language Independent Stemmer. I would like to express my sincere thanks to the Corpus Provider (EMILLE Corpus) who generously allowed me to download data from their website with free of cost.

REFERENCES

[1] Mohd. Shahid Husain, "*An Unsupervised Approach to Develop Stemmer*", In: International Journal on Natural Language Computing (IJNLC), Vol.**1**, Issue.**2**, pp.**15-23**, ISSN: **2278-1307**, **India**, 2012.

[2] Sajjad Ahmad Khan, WaqasAnwarl, UsamaIjaz Bajwa1, and Xuan Wang, "*A Light Weight Stemmer for Urdu Language: A Scarce Resourced Language*", In: Proceedings of the 3rd Workshop on South and Southeast Asian Natural Language Processing (SANLP), pp.**69–78**, **India**, 2012.

[3] Dhabal Prasad Sethi, "*Design of Lightweight Stemmer for Odia Derivational Suffixes*", In: International Journal of Advanced Research in Computer and Communication Engineering, Vol.**2**, Issue.**12,** pp. **4594-4597**, ISSN (Print): **2319-5940**, ISSN (Online): **2278-1021**, **India**, 2013.

[4] Sundar Singh, R K Pateriya, "*Enhanced Suffix Stripping Algorithm to Improve Information Retrieval",* In: International Journal of Computer Sciences and Engineering, Vol.**3**, Issue.**8**, pp.**115-119**, E-ISSN: **2347-2693**, **India,** 2015.

[5] Karaa WBA. "*A New Stemmer to Improve Information*", In: International Journal of Network Security and Its Applications (IJNSA), Vol.**5**, Issue.**4**, pp.**143–154**, ISSN: **0974-9330**, **India**, 2013.

[6] M. Kasthuri, Dr. S. Britto Ramesh Kumar, "*A Framework for Language Independent Stemmer Using Dynamic Programming*", In: International Journal of Applied Engineering Research (IJAER), Vol.**10**, Number.**18**, pp.**39000-39004**, Online ISSN: **1087-1090**, Print ISSN: **0973-4562**, **India**, 2015.

[7] M. Kasthuri, Dr. S. Britto Ramesh Kumar, "*PLIS: Proposed Language Independent Stemmer for Information Retrieval Systems Using Dynamic Programming*", In: 2016 World Congress on Computing and Communication Technologies, IEEE, pp.**132-135**, ISBN: **978-1-5090-5573-9**, **India**, 2016.

[8] M.Kasthuri, "*Proposed Architecture for Language Independent Stemmer*", In: International Open Access Journal of Emerging Technologies and Innovative Research (JETIR), Vol.**5**, Issue.**10**, pp.**943-948**, ISSN: **2349-5162**, October 2018.

[9] NavanathSaharia, KishoriKonwar, M., Utpal Sharma, and JugalKalita, K., "*An Improved Stemming Approach Using HMM for a Highly Inflected Language*", In: Springer-Verlag Berlin, ISBN: **9783642372476**, Vol.**7816**, Issue.**1**, pp.**164-173**, **Heidelberg**, 2013.

[10] Garima Joshi, and Kamal Deep Garg, "*Enhanced Version of Punjabi Stemmer Using Synset*", In: International Journal of Advanced Research in Computer Science and Software Engineering, Vol.**4**, Issue.**5**, pp.**1060-1065**, ISSN: **2277-128X**, **India**, 2014.

[11] Anshu Sharma, Rakesh Kumar and VibhakarMansotra, "*Proposed Stemming Algorithm for Hindi Information Retrieval*", In: International Journal of Innovative Research in Computer and Communication Engineering, Vol.**4**, Issue.**6**, pp. **11449-11455**, ISSN(Online): **2320-9801**, ISSN (Print): **2320-9798**, June 2016.

[12] Abebe Belay Adege, YibeltalChanieManie, "*Designing a Stemmer for Ge'ez Text Using Rule Based Approach*", In: International Journal of Scientific & Engineering Research, Vol.**8**, Issue.**1**, pp.**1574-1578**, ISSN: **2229-5518, Ethiopia**, 2017.

[13] Robert A. N. de Oliveira and Methanias C. Junior "*Experimental Analysis of Stemming on Jurisprudential Documents Retrieval*", In: Information, Vol.**9**, Issue.**28**, pp.**1-34**, **Brazil**, 2018.

[14] AttiaNehar, DjelloulZiadi, and HaddaCherroun, "*Rational Kernels for Arabic Stemming and Text Classification*", In: Springer-Verlag Berlin Heidelberg, Vol.**1**, Issue.**1**, pp. **176–187**, **Algerie, 2015**.

[15] M. Kasthuri, Dr. S. Britto Ramesh Kumar, "*PLIS: Proposed Language Independent Stemmer Performance Evaluation*", In: International Journal of Advanced Research in Computer Science & Technology (IJARCST), Vol.**5**, Issue.**4**, pp.**943-948**, ISSN: **2347-8446** (Online), ISSN: **2347-9817** (Print), **India**, 2015.

[16] Gunadeep Chetia, Gopal Chandra Hazarika, "*Pre-processing Phase of Automatic Text Summarization for the Assamese Language*", In: International Journal of Computer Sciences and Engineering, Vol.6, Issue.**10**, pp.**159-163**, E-ISSN: **2347-2693**, **India,** 2018.

[17] Dharmendra Sharma, Suresh Jain, "*Evaluation of Stemming and Stop Word Techniques on Text Classification Problem*", In: International Journal of Scientific Research in Computer Science and Engineering, Vol.**3**, Issue.**2**, pp.**1-4**, ISSN: **2320-7639**, India, 2015.

[18] John Bosco.P, S.K.V.Jayakumar, "A Study on Web Based Image Search by Re-Ranking Techniques", In: International Journal of Scientific Research in Network Security and Communication, Vol.**5**, Issue.**3**, pp.**19-26**, ISSN: **2321-3256**, India, 2017.

Authors Profile

M.Kasthuri is working as an Assistant Professor in the Department of Computer Applications, Bishop Heber College, Tiruchirappalli, Tamil Nadu, India. She had completed her Doctorate of Philosophy in Computer Science in June 2017 at Bharathidasan University, Tiruchirappalli. She has published a number of National and International level research papers related to Web Mining and Stemming concepts. She has completed UGC sponsored Minor Research Project entitled as Language Independent Stemmer.

