Indian Journal of Science and Technology, Vol 9(12), DOI: 10.17485/ijst/2016/v9i12/86631, March 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Maulik: A Plagiarism Detection Tool for Hindi Documents

Urvashi Garg* and Vishal Goyal

Punjabi University, Patiala, NH 64, Urban Estate Phase II, Patiala - 147002, Punjab, India;
urvashi.mittal80@gmail.com, vishal.pup@gmail.com

Abstract

Objective: The objective of this paper is to present an automated plagiarism detection software tool called Maulik. There are many plagiarism detection tools available for English text. Maulik detects plagiarism in Hindi documents. **Method:** Maulik divides the text into n-grams and then matches it with the text present in repository as well as with documents present online. Preprocessing techniques such as stop word removal and stemming has been used. The best value of n-gram for finding out the similarity of two Hindi documents has also been found out. Cosine similarity has been used for finding the similarity score. **Findings:** Similarity score of 96.3 has been achieved which is higher as compared to the existing Hindi plagiarism detection tools such as Plagiarism checker, Plagiarism finder, Plagiarisma, Dupli checker, Quetext. These tools compared only exact matches ignoring the language specific constraints whereas Maulik is capable of finding plagiarism if root of a word is used or a word is replaced by its synonyms. **Application:** Maulik is a software tool which discourages plagiarism as well as motivates the writing skills of people.

Keywords: Cosine Similarity, Plagiarism, Stemming, Stop Word, Synonyms

1. Introduction

Plagiarism is a serious and widespread educational issue. The Oxford Dictionary meaning of Plagiarism is the practice of taking someone else's work or ideas and passing them off as one's own. According to **Office of Student Judicial Affairs** "Plagiarism means using another's work without giving credit. If you use others' words, you must put them in quotation marks and cite your source. You must also give citations when using others' ideas, even if you have paraphrased those ideas in your own words"[1]. The word Plagiarism is derived from Latin roots: *plagiarius,* an abductor, and *plagiare,* to steal. According to, **American Historical Association** "The expropriation of another author's work, and the presentation of it as one's own, constitutes plagiarism and is a serious violation of the ethics of scholarship"[2]. Plagiarism also includes:

- Turning in someone else's work as your own.
- Copying words or ideas from someone else without giving credit.
- Failing to put a quotation in quotation marks.
- Giving incorrect information about the source of quotation.
- Changing words but copying the sentence structure of a source without giving credit.
- Copying so many words or ideas from a source that it makes up the majority of your work, whether you give credit or not.

It is very hard to give any exact definition of plagiarism as it is not possible to analyze the meaning of natural languages thoroughly. Definitions studied are[3]:

Definition 1.1 *Two sentences are v-similar if they contain the same sequence of* v *consecutive words.*

Definition 1.2 *Two documents* D *and* D' *are* (v,m)*- similar if there exist* m *sentences* $s_1, \ldots, s_m$ *in* D, *and* m *sentences* $s_1', \ldots, s_m'$ *in* D', *such that* s_i *is* v*-similar to atleast one* s_j' *for* $j \in \{1, \ldots, m\}$. We do not require that these sentences to be consecutive in either of the documents.

*Author for correspondence

The University of North Carolina has given a handout which gives complete information on plagiarism. It gives the steps which can be followed to avoid plagiarism.

At UNC, plagiarism is defined as "the deliberate or reckless representation of another's words, thoughts, or ideas as one's own without attribution in connection with submission of academic work, whether graded or otherwise". (*Instrument of Student Judicial Governance,* Section II.B.1.). Because it is considered a form of cheating, the Office of the Dean of Students can punish students who plagiarize with course failure and suspension. The handout[4] gives a description of what actually plagiarism is and how plagiarism can be avoided.

According to UNC, plagiarism can be avoided by using citations even if

- All direct quotes have been put in quotation marks.
- Synonyms have been used.
- Paraphrasing of the ideas to which you referred have been done.
- Sentence is mostly made up of your own thoughts, but contains a reference to the author's ideas.
- Author's name has been mentioned in the sentence.

Plagiarism may arise in written text (articles, books, newspapers, e-mail messages), source code (various programming languages), visual text (paintings, photographs), spoken text (speech, lectures), multimedia products (movies, presentations), music (CD, MP3), etc. Now a days, written text is the main challenge we face. There are many institutions (education, research, politics) where originality of work or ideas is to be checked. Many Universities have enforced plagiarism policy and plagiarism detection software to discourage act of plagiarism.

2. Related Work

In[5] presented a tool called Sif for finding similar files. Fingerprint approach is used in Sif. Sif works in two modes one against all and all against all. The approach used in sif is completely syntactic means files containing similar information but different words were not considered. The method is susceptible to bad fingerprints. The space required to store fingerprints is 5% of total space for all against all. A better encoding can reduce it to 2%.

Turnitin[6] is a creation of iParadigms. It uses a database of student papers to identify plagiarism. It is a paid tool and is available online. Registered users can either upload whole document to be checked to the site or use copy-and-paste sections. Some proprietary algorithms are used for calculating the fingerprints of the document, and then comparing it to the other fingerprints in several very large scale databases. Turnitin claims to be able to detect fraud when lexis has been altered. Proprietary algorithms are used to search three main sources: first is the current and extensively indexed archive of Internet with approximately 4.5 billion pages, second is books and journals in the ProQuest™ database; and the last is 10 million documents submitted in the Turnitin corpus. A Sentence based natural language plagiarism detection algorithm which has been integrated in Sherlock[7]. Earlier it was used to detect programming plagiarism but presently it is used to detect source code as well as natural language plagiarism. Complexity for algorithm is $O(n^2)$. Variations of various parameters like common threshold, similarity threshold and common word lists is done. It compares all the text in current directory and produces a listing of the most similar files, together with a percentage similarity index. Sherlock has good quality of results and method of result visualization. It is a command line program and it doesn't have a graphical interface. RoboProf[8] is an internal plagiarism detection system. It generates and evaluates assignments as well as provides comment to students. In its plagiarism detection module, it works by adding a watermark to student's assignment at submission time. If someone else gets the electronic copy of the same program and submits it, RoboProf detects it. The watermark is a binary number formed by concatenating the year, exercise id, student id and a checksum. RoboProf is language independent. One major advantage of RoboProf is it detects copied work from previous years also. Major disadvantage is it can detect only if electronic copy of program is taken. Also, student can disturb the watermark. The tool[9] is language sensitive and works for cases which has stylometric approach. They adapted Morton and Hilton work to produce 62 measurements. In future stylometric heuristics can be combined with other methods to improve the results. [10]Linguistic information was used to identify paraphrased documents. They used Levin's semantic verb classes to describe the expression. Their experiments used translated novels as surrogate for plagiarism data[11] presented algorithms for pattern matching which works for single and multiple patterns. They presented Basic Parameterized Shift-or algorithm (PSO), Fast Parameterized Shift-or algorithm (PFSO) and Parameterized Backward Trie Matching (PBTM).

They showed PSO and PBTM gives the best results for short pattern. PFSO gives good results for long patterns. They showed already existing algorithms Boyer-Moore and Aho-Corasick automation algorithms are worse than theirs. All previous algorithms prove optimal worst case time whereas theirs have good average case time. EVE2[12] is a plagiarism detector called the Essay Verification Engine. EVE2 is a very powerful tool that detects plagiarized material from the World Wide Web. EVE2 accepts documents in plain text, Microsoft Word, or Corel Word Perfect format. Eve performs a large number of complex searches to find the suspected site. Then it does a direct comparison of the submitted essay to the text appearing on the suspect site. If it finds traces of plagiarism, the URL is saved. The results include similarity index with copied text highlighted in red. It also gives URLs from which text has been copied. SNITCH[13] (Spotting and Neutralizing Internet Theft by Cheaters) is a software tool which employs a sliding window technique to scan a document and locate candidate passages that might be plagiarized. Each passage is searched for on the Internet. It is a tool for detecting cut and paste plagiarism. The authors also compared their tool with EVE2. They both have similar format for output. They showed SNITCH as an improvement over EVE2 on the measures of time and accuracy. A new two-step approach[14] is introduced for plagiarism detection. They proposed the blend of high detection speed of Fast Plagiarism Detection System with enhanced reliability and reporting potential of Plaggie. They worked on hermetic plagiarism detection systems and used content based comparison. They showed combined system is much faster than Plaggie and also FPDS as a filter does not clearly reduce the reliability of Plaggie.

Glatt[15] is also an internal plagiarism detection system, like RoboProf. This tool works on the writing styles of a human being. It works by substituting every fifth word of a text with a blank space. Then the presenter of the text is asked to replace the missing words. Proportion of plagiarized text is found from the incorrect substituted words. This manual process makes the method least suitable for larger texts. Moreover, there is possibility that a plagiarist can pass this test by rote learning of the original text. Plagiarism-Finder[16] compares the given document with documents available on the Internet. Installation of the tool on system is required. It works on Windows 2000 and XP systems and recognizes files in several standard formats such as PDF, DOC, HTML, TXT and RTF. It outputs HTML reports highlighting plagiarized text and providing URLs to the source. iThenticate[17] in addition to comparing a given document against the sources available on the World Wide Web, it also compares the given document against proprietary databases of published works as well as various electronic books. The originality reports provide the quantity of materials copied to determine the extent of plagiarism. It doesn't require installation on home computer. Ephorus[18] requires no installation on the system. The system can be freely tried. It only requires registration with the site. Documents are uploaded on the Ephorus website. The search engine compares the given document to the documents on the internet and gives an originality report. PlagiarismDetect[19] is a freely accessible online tool. Users need to register by providing their names and email addresses. Once registered, passage can be entered in the text box provided or a file can be uploaded for study. A description is then sent back to the user with a list of the URLs from where the text has been copied with fraction referring to the amount copied. Urkund[20] is a server based plagiarism detection web service. It uses standard email systems for submission of documents and viewing results. This tool search through all available online sources and previous published work also. The originality report consists of highlighted copied text and URLs from where the text has been copied. CopyCatch Gold[21] is used for collusion detection. It has a large number of commercial users. It offers a CopyChecker which is free with CopyCatch Gold which teaches students about how to use sources appropriately. It also offers a web version which broadens the competence of plagiarism detection across the internet using the Goggle API. WCopyfind[22] is an open source hermetic tool for detecting plagiarism against a local repository of documents. The product is being extended to search across the internet net using the Google API. This is a free online available tool. Doc Cop[23] is a web based plagiarism detection system for written text. It accepts both doc and pdf files. When the files are checked against one another or the Web, DOC Cop allows selection of string length. It gives reports in HTML or doc formats. It claims to have fast turn around time. Anti-twin[24] is a feely available system which checks the plagiarism in local corpus only. It needs to be downloaded on the system. Results are not pleasing. CHECK[25] uses structural information and keywords from the documents as features to check the overlap. It works well for non-reworded documents. Instantaneous results are available with a similarity index. Plagiarism Checker[26]

is a plagiarism detection tool which is freely available online. The tool gives instantaneous results but results are not appropriate. The tool searches the web for the duplicate content and detects if the sentences matches exactly. Plagiarisma[27] is a freely available online desktop application. The tool supports 190 languages including Hindi, Nepali, Sanskrit etc. The tool is able to detect exact matches only that too not very appropriate. The Synonymizer tool makes plagiarism possible by rewriting sentences with synonyms to produce unique text. Even registered users cannot scan documents for more than 3 times per day. ACNP Software[28] includes two modules AntiPlagiarist and AnticutAndPaste. The tool supports English language only. It requires no uploading to the external servers. AntiCutAndPaste is designed to search for text that have been copied and pasted in programming language source code or plain text. It is tested on sources from large C++, Visual Basic, Delphi, Java, and C# projects. Plagium[29] is a paid plagiarism detection software available online. The tool works well and gives very fast response. It works for English language only. Dupli checker[30] is a 100% free online plagiarism detection tool and is extremely easy to use. It has the 3 options for entering the input. By copy-pasting the text, entering the URL of the text required to be checked, or uploading a text file. The number of searches is limited to 50 searches per day for registered users and only 1 search per day for unregistered users. The report doesn't give similarity score and detects exact matches only. PaperRater[31] is a paid tool. It Offers 3 tools: Grammar checking, plagiarism detection, and writing suggestions. It is developed and maintained by linguistics professionals and graduate students. Reports are not saved. In paid version it accepts longer documents (up to 6000 words). It doesn't work for non-English documents. Quetext[32] is 100% free and has an easy to use interface. There is no need to download software or create any account. Text can be only copy and paste. There is no option for uploading files. It works for Hindi text but doesn't give accurate results. Viper[33] is also 100% free software for plagiarism detection tool. It requires a download on the system. It also doesn't work for Hindi text. For English text too, the results are not satisfactory. Plagiarism checker X[34] tool needs to be downloaded on the system. Free for the users. It doesn't work for Hindi text. PlagScan[35] is an online available tool gives 20 free trials to the users. It generates reports to compare the query document with matched documents. It doesn't work for Hindi language.

3. Methodology and System Architecture

There are various plagiarism detection tools available for processing text in different languages. But none reveal their methodology and architecture[5,7,14] are Hermetic systems which check the documents in its registered corpus only whereas[6,16,20] are the software which checks internet resources for plagiarism[16,26,27,30,32]. Work for Hindi text but don't consider insertion or deletion of irrelevant words. Also if a word is substituted by its synonym they don't consider it as plagiarism. Maulik is an open system for written text which checks its corpus as well as internet documents also. Mostly tools works on the syntactic information of the text[24]. Maulik is structural system which considers partial understanding of documents. Because it is not mere content based comparison[5,24], it is language dependent and works for Hindi documents only.

The System has following modules:

1. Stop Word Removal
2. Stemming
3. Conversion into N-gram
4. Synonym Replacement
5. Calculation of tf-idf and Similarity Score

The measureless intricacy of the Natural Language jointly with space and processing time constraint of the computers has raised the requirement for the reduction of the amount of information to be processed by the computational algorithms. This requirement showed the way to the design of preprocessing techniques that reduced the linguistic input while loosing the least amount of semantic information. These preprocessing techniques not only paced up the algorithms but they even enhanced their performance in many NLP tasks[36]. The preprocessing techniques which are taken up in this architecture are explained in 3.1 and 3.2.

3.1 Stop Word Removal (SWR)

It is a fundamental pre-processing approach that removes common words. Its primary use is to prevent the following processing being over-influenced by very frequent words. Such words include articles, prepositions, and other function words. These are rather noise, which might decrease the accuracy, and therefore it is preferred to remove them.

The two main reasons for removing repeatedly occurring words having no factual use in describing article contents during plagiarism detection. First, each match between a suspected document and a query document should be based on relevant terms rather than simply because they contain words such as मुझे, आपको, मैने, तू. It would not constitute an intelligent strategy since these non significant words in fact represent noise and may actually damage performance because they do not discriminate between relevant and irrelevant items. Second, the number of n-grams to be compared also gets reduced. Stop words are the least content bearing words. Stop word elimination can be considered as an implementation of Zipf's law, where high frequency terms are dropped from a set of index terms[37]. A list of stop words has been taken from[38]. Hindi documents from EMILLE[39] corpus have been used for finding out the stop words. UTF-8 encoding is used.

Algorithm followed:

Input: A list of stop words s[]
Hindi text
Output: An array c consisting of text without stop words.
Initialization: x=0

- *Input the text.*
- *Tokenize the text into words[].*
- **for** *each i in words[]*
  for *each j in s[]*
  if *words[i] =s[j]*
  go to step 3
  else
  x = 1

if *(x = 1)*
put the word in c[] and increment the index

Frequency of each word in the corpus is calculated. The corpus consists of approximately 60.4 million words out of which there are 1.24 million unique words. Many content bearing words also appeared with high frequency. After analysing the 3000 words having highest frequency, a list of 205 Hindi stop words has been created. E.g. मुझे, आपको, मैने, तू, तुम्हे, तुम, इत्यादि, इन्हीं, उन्हीं, वगैरह, लिये etc. When stop words are removed from a corpus in the system, it is found that out of 22.6 million words, the frequency count of stop words is 8.9 million which covers approximately 40% of corpus. So removal of the stop words reduces the size of the corpus significantly. Reduction in size of corpus leads to less number of n-grams and less number of index terms. Along with stop words, special characters like | , ! etc. are also removed.

3.2 Stemming (STR)

Stemming is the process of determining the base form of a given word. During this process, the context of the word is used to determine the word sense. This can then be used to select an appropriate base form. Stemming is the process for reducing inflected words to their stem. The main purpose of stemming is to reduce different grammatical forms/word forms of a word like its noun, adjective, verb, adverb etc. to its root form. Stemming is widely uses in Information Retrieval system and reduces the size of index files. The goal of stemming is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. Many stemmers have been developed for different languages using different approaches.

This is a light weight stemmer which works on Suffix stripping with dictionary look up.

Algorithm is given below:

Input: L is a suffix list which consists of a list of Hindi suffixes for stripping.
C: Hindi Corpus
Output: root words of text.

- *Input the text.*
- *Tokenize the text into words*
- *For each W in text*
  for *each suffix (suff) present in L*
  if *(W ends with suff and (suff.length < W.length))* **then**
  put the root words in a[] after suffix stripping

else
root = W
go to step 4
for *each(a[i])*
check the frequency of each a[i] in corpus.
Put a[i] with maximum frequency in r.

If *(frequency r>4)*
root= r
go to step 4.
else
add "ा", "ी" to r.
put the resultant words in S[].
for *each S[i]*
Check the frequency of each S[i] in corpus.
Put S[i] with maximum frequency in r.
If*(frequency of S[i]>4)*
root= S[i]
else root=W
go to step 4

- *Output is in root*

A list of suffixes with their frequencies has been created. There is large number of Hindi words and hence the extraction of suffixes formed a large list. Out of approximately 2000 identified suffixes, 148 suffixes with highest frequencies have been used for stripping.

After stripping the suffix of a word, the word is checked whether it is relevant or not by checking it in the corpus. If it is not relevant, "ा", "ी" are added. The reason for doing so is mentioned below:

When अलमारियाँ is stripped to find its root we get अलमार. Because it is an irrelevant word, it wont be available in the corpus. To find a relevant root ा and ी are added to it, which gives us अलमारा and अलमारी. Now checking these words in corpus gives us अलमारी as root.

3.3 Conversion into n-grams

In the fields of computational linguistics and probability, an n-gram is a contiguous sequence of n items from a given sequence of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The n-grams typically are collected from a text or speech corpus. An n-gram of size 1 is referred as unigram, size 2 as bigram, size 3 as trigram and so on[40]. The extraction process utilizes the sliding window principle to retrieve all N-grams of the specific length.

To calculate the number of n-grams in a text:

$$|N| = w - (m - 1)$$

Here, w is number of words in the text, m is the value on n in n-grams.

The purpose of using n-grams for plagiarism detection is given below. N:N matching has been used.

Consider the following example:

हिन्दी संवैधानिक रूप से भारत की प्रथम राजभाषा है

Considering as trigrams the sentence can be represented as:

हिन्दी संवैधानिक रूप(1), संवैधानिक रूप से(2), रूप से भारत(3), से भारत की(4), भारत की प्रथम(5), की प्रथम राजभाषा(6), प्रथम राजभाषा है(7)

Total n-grams = 7

1. Inserting new word

   हिन्दी संवैधानिक रूप से भारत देश की प्रथम राजभाषा है

   हिन्दी संवैधानिक रूप(1i), संवैधानिक रूप से(2i), रूप से भारत(3i), से भारत देश(4i), भारत देश की(5i), देश की प्रथम(6i), की प्रथम राजभाषा(7i), प्रथम राजभाषा है(8i)

   Matching (1|1i) (2|2i) (3|3i) (6|7i) (7|8i)

2. Deletion of a word

   हिन्दी संवैधानिक रूप से भारत की राजभाषा है

   हिन्दी संवैधानिक रूप(1d), संवैधानिक रूप से(2d), रूप से भारत(3d), से भारत की(4d), भारत की राजभाषा(5d), की राजभाषा है(6d)

   Matching (1|1d) (2|2d) (3|3d) (4|4d)

3. Substituting a word in the text

   हिन्दी संवैधानिक रूप से हिंदुस्तान की प्रथम राजभाषा है

   हिन्दी संवैधानिक रूप(1s), संवैधानिक रूप से(2s), रूप से हिंदुस्तान(3s), से हिंदुस्तान की(4s), हिंदुस्तान की प्रथम(5s), की प्रथम राजभाषा(6s), प्रथम राजभाषा है(7s)

   Matching (1|1s) (2|2s) (6|6s) (7|7s)

As shown in above example, one word insertion, deletion or substation can cause maximum 3 mismatches (value of n). Where as if we compare the whole sentence as such no matching occurs. Hence it is better to divide the text into n-grams and then process it.

3.4 Synonymy Recognition (SYR)

The motivation for using synonymy recognition comes from considering human behavior, where people may seek to hide plagiarism by replacing words with appropriate synonyms. It visually changes sentences at first sight, but the structure is left unmodified. If a sufficient number of words are replaced by synonyms, then most of the common copy detection methods fail. Regardless of the features the methods use, the best solution is to transform words having the same or closely related meaning onto a unique identifier. Hindi WordNet from IIT, Bombay[41] has been used in this work to find the synonyms. It consists of synonyms of approximately 38000 words and total words are approx 1,500,00.

Algorithm used is given below:

Input: n-gram, Hindi Synonym dictionary, IDF of n-grams in corpus

- *Input the n-gram.*
- *If n-gram is not found in the repository*
  Generate synonyms of first word of n-gram
  Process all the synonyms one by one
  If the generated synonym n-gram is found in repository
  Find and replace synonym in matched n-gram list
  Else add synonym n-gram in list of non-matched n-grams.

*Calculate TF * IDF*

3.5 Calculation of tf-idf and Similarity Index

The problem of finding the similarity has been studied broadly. Many measures are available. In Maulik, Cosine similarity[41] has used for calculation of similarity of query document with documents present in corpus. In this

method, a weight is assigned to each term according to its importance in a particular document. A commonly used term weighting method is tf-idf, which assigns a high weight to a term, if it occurs frequently in the document but rarely in the whole document collection. Contrarily, a term that occurs in nearly all documents has hardly any discriminative power and is given a low weight, which is usually true for stop words. For calculating the tf-idf weight of a term in a particular document, it is necessary to know two things:

Term frequency (tf): How often does a term occur in the document.

Document Frequency (dfi): In how many documents of the collection does it appear.

The inverse document frequency (idf) is calculated as the logarithm (base 10) of the quotient of the total number of documents (N) and the document frequency in order to scale the values. Now both the values which are required for calculating weights are available which is tf*idf. Vetcor length is calculated as:

$$|D| = \sqrt{\sum_{i} (w_{i,j})^2}$$

Then the dot products of Query vector with document vector is calculated.

$$Q \bullet D_i = \sqrt{(w_{Q,j} * w_{i,j})}$$

Cosine value is calculated as:

$$\cos\theta(d) = \frac{(Q \bullet d)}{|Q| * |d|}$$

3.6 Architecture

The complete architecture is given in Figure 1.

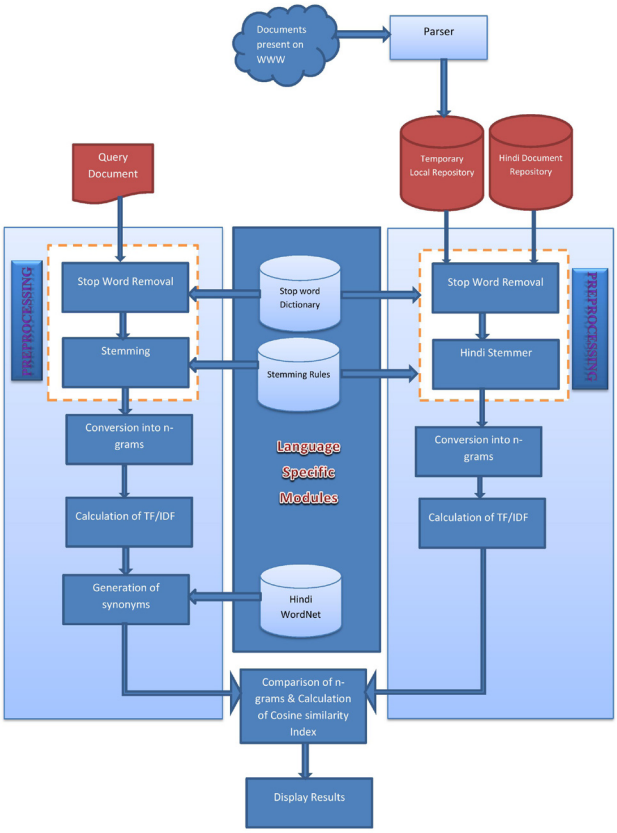

Figure 1. Architecture of Maulik.

4. Experiments & Results

For experiments, similarity of vectors of tf-idf weighted n-grams has been used.

A corpus has been created in the system which consists of approximately 100 documents. Maulik verifies the documents present in the local repository as well as present on WWW. In this web based information retrieval system, documents are collected from the WWW on the basis of keywords and the title of the paper. A temporary local database is created in the system for calculating the similarity of query document with the retrieved parsed documents present in local database. The temporary database is flushed out after the process of calculating similarity score is complete. Different experiments have been performed to check the effect of various modules on similarity of documents and hence plagiarism. Tf-idf values have been calculated and stored in the corpus for use. If the value of n in n-gram is not changed tf-idf need not to be calculated again. But if the value of n in n-grams is changed tf-idf is recalculated which means it will take more time to calculate similarity.

4.1 Selection of Value of n-gram

Similarity index has been find out varying the value of n-grams. Experiments reveal that with value greater than 5 degrades the performance of the system in terms of similarity as shown in figure below. Also working on unigram or bigram treats the text as a bag of words and hence doesn't contribute to the similarity of two documents. For a low value of n, the number of n-grams in the document will be large and hence processing will take more time. So comparing processing time and accuracy, value of 5 is taken as appropriate value of n in n-gram, which is illustrated in Figure 2.

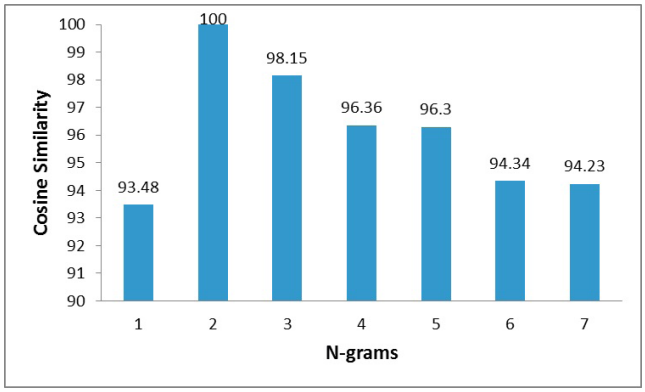

Figure 2. Variation in Cosine Similarity with change in n-grams.

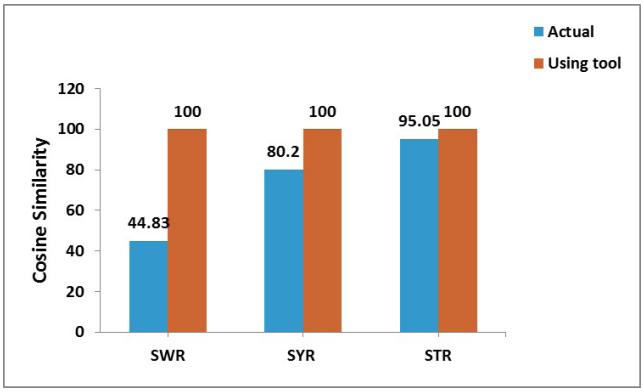

Figure 3. Variation in Cosine Similarity with and without using tools.

4.2 Effect of Stop Word Removal, Stemming and Synonym Replacement on Cosine Similarity

Various experiments have been performed to show the effect of Stop Word Removal, Stemming and Synonym Replacement on similarity of text between documents. 10 documents of approximately 1100 words in each are taken. For the first experiment, some of the Stop words have been removed from the document and then similarity is calculated between the original document and the modified document. Similarity is calculated with and without stop word removal module. The results show that for calculating the similarity of two documents, if stop words are removed from both of them, similarity is increased. Because stop words are not content bearing words, hence it is better to remove them. Similar kind of work has been done to see the effect of stemming on similarity of documents. Some of the words has been stemmed and compared against the original document. Use of stemmer improves the similarity score. For synonym replacement, significant improvement can be seen if Hindi WordNet is used for finding the synonyms. Hence implementation of each module increases the similarity score and improves the performance of the system. Comparison of similarity scores is given in Figure 3.

For second experiment, effect of two modules have been have been analysed on the text. Some Stop Words have been removed and some words have been stemmed in the document. Result of implementation of modules can be seen from the Figure 4 given below. Stop word Removal and Synonym Replacement shows significant improvement in the similarity of text. When no tool is used for calculating the cosine similarity the score comes out be 43.68 where as when both tools are used, it raised to 92.59 which is a significant improvement.

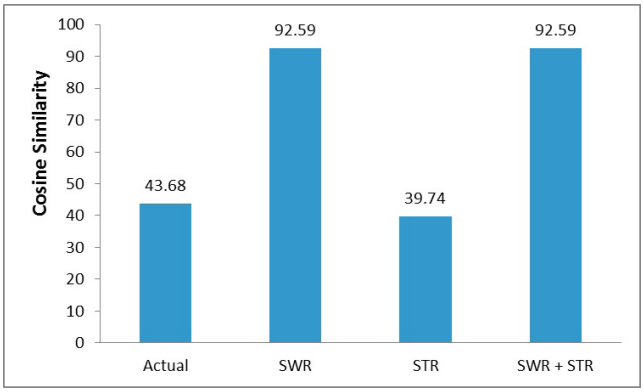

Figure 4. Effect of Stop Word Removal + Stemmer on Similarity of text.

Similarly, improvement in the performance after using Stop Word Removal + Synonym Replacement and Synonym Replacement + Stemmer can be analysed from the Figure 5 and Figure 6.

Finally, the effect of all three modules is analysed on the text is shown in Figure 7. The text contained the synonyms, stemmed words and also some stop words were removed. Reults shows using all the three modules increases the similarity score.

4.3 Comparsion with other Available Tools

There are various desktop and online tools available for plagiarism detection. But a very few works for Hindi text. A small piece of hindi text was given to the available tools working for Hindi language. Some words were replaced by their synonyms. The result is as shown in Table 1:

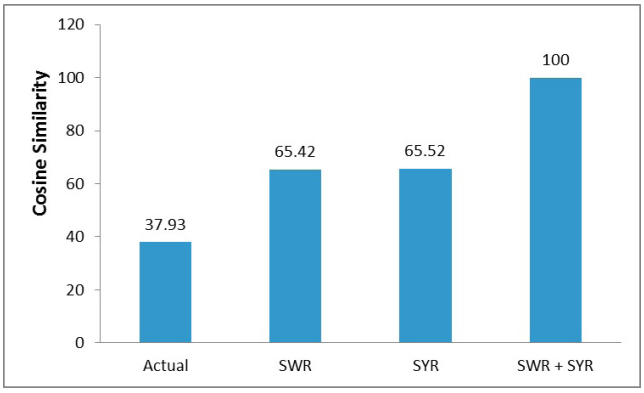

Figure 5. Effect of Stop Word Removal + Synonym Replacement on Similarity of text.

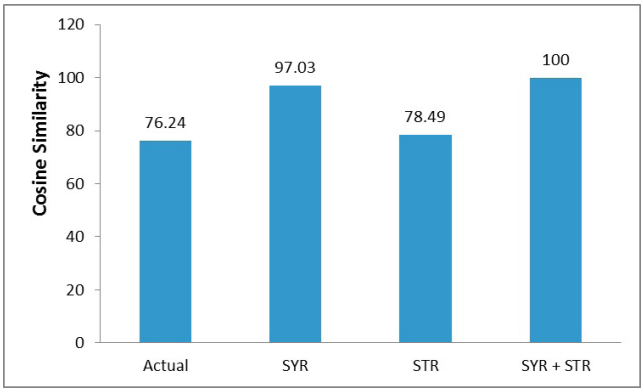

Figure 6. Effect of Synonym Replacement + Stemmer on Similarity of text.

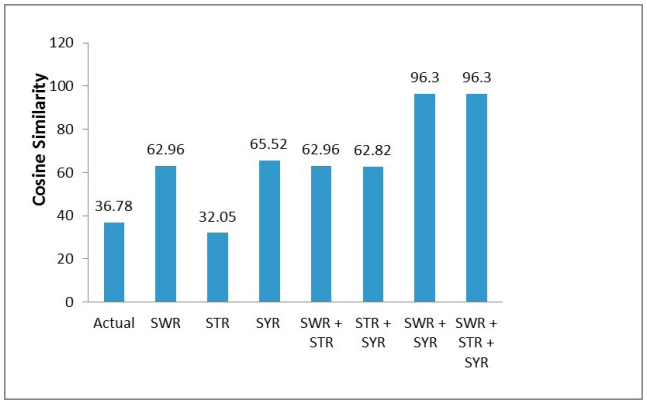

Figure 7. Effect of Stop Word Removal + Synonym Replacement + Stemmer on Similarity of text.

Various other famous plagiarism detection tools like Turnitin, WCopyFind, Ithenticate, COPS don't work for Hindi text. The snapshots of different tools are shown in Figure 8, Figure 9, Figure 10, Figure 11, Figure 12, Figure 13.

Table 1. Comparison of various plagiarism detection tools for Hindi text

Sno.	Name of the tool	Similarity Index
1	Plagiarism checker	0%
2	Plagiarism finder	77.5%
3	Plagiarisma	60%
4	Dupli checker	-
5	Quetext	0%
6	Maulik	96.3%

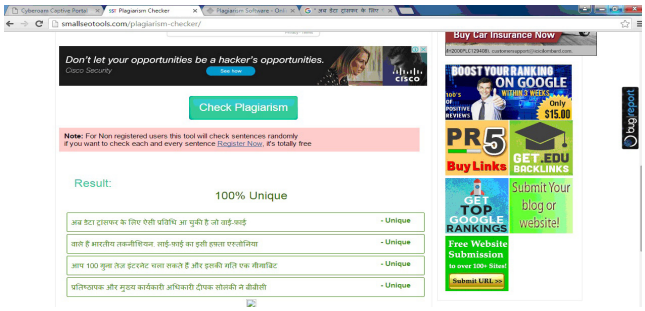

Figure 8. Plagiarism Checker.

Figure 9. Plagiarism Finder.

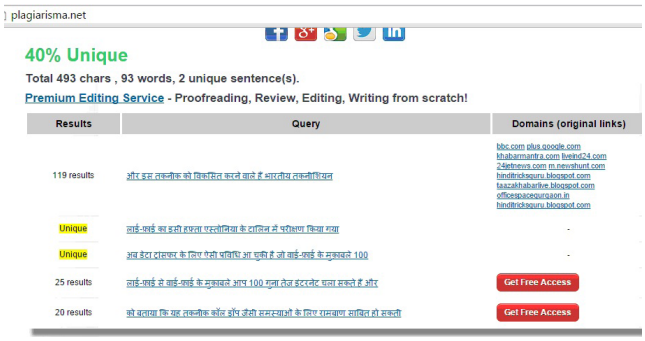

Figure 10. Plagiarisma.

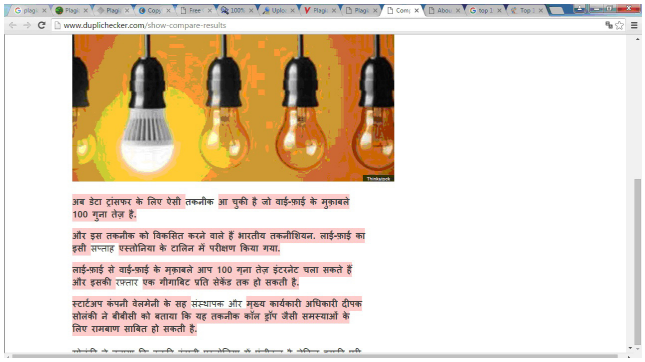

Figure 11. Dupli checker.

Figure 12. Quetext

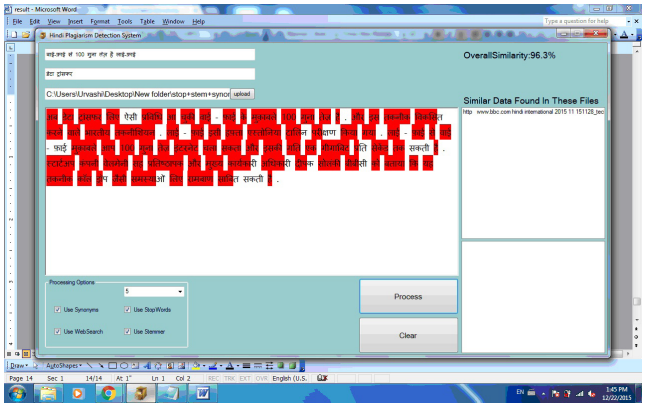

Figure 13. Maulik

5. Conclusion

Plagiarism detection is a complex task and many issues are closely related to it. A wide range of combinations of text pre-processing techniques have been explored in terms of the quality of plagiarism detection. Numbers of plagiarism detection tools have been developed for documents containing English text but very less work has been done for Hindi text. As shown above the tools which have been already developed for plagiarism detection for Hindi text works only for finding the exact copy. Basically, they match the strings only. If some words are replaced with their synonyms or root words are used, the tool fails. A general approach to hide plagiarism is by restructuring the sentences by using synonyms or root words or change in stop words. The tool Maulik is capable of finding similarity in the above mentioned cases also. Undoubtedly, it is not possible for any tool to work as a human brain but this plagiarism detection system ascertains its performance to an acceptable level.

6. References

1. Avoiding Plagiarism. Date Accessed: 5/09/2011: Available from: http://sja.ucdavis.edu/files/plagiarism.pdf.
2. American Historical Association: Statement on Standards of Professional Conduct. Date Accessed: 12/07/2011: Available from: http://www.historians.org/pubs/free/ProfessionalStandards.cfm.
3. Sorokina D, Gehrke, J, Warner S, Ginsparg P. Plagiarism Detection in arXiv. Hong Kong, China: Proceedings of the 6th International Conference on Data Mining. 2006; p. 1070-75.
4. The Writing Center. Date Accessed: 15/11/2014: Available from: http://writingcenter.unc.edu/handouts/plagiarism/.
5. Manber U. Finding Similar files in a Large File System. San Francisco, California: Proc. of USENIX. 1994; p. 1-10.
6. Turnitin: Leading Plagiarism Detection Tool. Date Accessed: 10/09/2011: Available from: http://www.turnitin.com.
7. White D., Joy M. Sentence-based Natural Language Plagiarism Detection. ACM Journal of Educational Resources in Computing. 2004; 4(4):1-20.
8. Daly C, Horgan J. Patterns of Plagiarism. Proceedings of the 36th SIGCSE Technical Symposium on Computer Sc. Education. 2004; 37(1):383-87.
9. Gruner S, Naven S. Tool Support for Plagiarism Detection. USA: Proceedings of ACM symposium on Applied Computing. 2005; p. 776-81.
10. Uzuner O, Katz B, Nahsen T. Using Syntactic Information to Identify Plagiarism. USA: Proc. 2nd Workshop on Building Educational Applications using NLP. 2005; p. 37-44.
11. Fredrisson K, Mozgovoy M. Efficient Parameterized String Matching. Journal of Information Processing. 2006; 100(3):91-96.
12. Essay Verification Engine, EVE Plagiarism Detection System. Date Accessed: 12/07/2011: Available from: www.canexus.com/eve/index.shtml.
13. Niezgoda S, Way TP. SNITCH: A Software Tool for Detecting Cut and Paste Plagiarism. SIGCSE '06 Proceedings of the 37th SIGCSE Technical Symposium on Computer science education. 2006; 38(1):51-5.

14. Mozgovoy M, Karakovskiy S, Klyuev V. Fast and Reliable Plagiarism Detection System. Proc. of FIE'07 Conf. 2007.
15. Glatt Plagiarism Services. Date Accessed: 12/08/2011: Available from: http://www.plagiarism.com.
16. Plagiarism Finder. Date Accessed: 10/08/2014: Available from: http://nbridge-plagiarismfinder.en.softonic.com/download#downloading.
17. iThenticate. Date Accessed: 10/08/2011: Available from: www.ithenticate.com/static/home.html.
18. Ephorus. Date Accessed: 12/08/2011: Available from: www.ephorus.de/home_en.html.
19. PlagiarismDetect. Date Accessed: 13/08/2011: Available from: www.plagiarismdetect.com.
20. Urukund. Date Accessed: 20/08/2014: Available from: www.urkund.com.
21. CopyCatch Gold. Date Accessed: 15/07/2014: Available from: http://www.cflsoftware.com/GoldFull.html.
22. Wcopyfind. Date Accessed: 20/01/2015: Available from: http://plagiarism.bloomfieldmedia.com/z-wordpress/software/wcopyfind/.
23. Doccop. Date Accessed: 20/08/2015: Available from: https://www.doccop.com/index.html.
24. Anti Twin. Date Accessed: 20/09/2015: Available from: http://www.anti-twin.com/.
25. Si A, Leong H, Lau R. CHECK: A Document Plagiarism Detection System. In Proceedings of ACM Symposium for Applied Computing. February 1997; p. 70-77
26. Plagiarism Checker. Date Accessed: 17/11/2015: Available from: http://smallseotools.com/plagiarism-checker/.
27. Plagiarisma. Date Accessed: 17/11/2015: Available from: http://plagiarisma.net/download.php.
28. ACNP Software. Date Accessed: 17/11/2015: Available from: http://www.anticutandpaste.com/antiplagiarist/.
29. Plagium. Date Accessed: 17/11/2015: Available from: http://www.plagium.com/.
30. Dupli Checker. Date Accessed: 18/11/2015: Available from: http://www.duplichecker.com/.
31. PaperRater. [Cited 18th November, 2015] Date Accessed: 18/11/2015: Available from: http://www.paperrater.com/.
32. Quetext. Date Accessed: 18/11/2015: Available from: http://www.quetext.com/.
33. Viper. Date Accessed: 19/11/2015: Available from: http://www.scanmyessay.com/plagiarism/.
34. Plagiarism Checker X. Date Accessed: 20/11/2015: Available from: http://plagiarism-checker-x.en.softonic.com/download#downloading.
35. PlagScan. Date Accessed: 21/11/2015: Available from: http://www.plagscan.com/.
36. Pandey AK, Siddiqui TJ. Evaluating Effect of Stemming and Stop-word Removal on Hindi Text Retrieval. Proceedings of the First International Conference on Intelligent Human Computer Interaction. 2009; p. 316-26.
37. Manning CD, Schutze H. Foundations of Statistical Natural Language Processing. England: The MIT press Cambridge. 1999; p. 23-24.
38. Garg U, Goyal V. Effect of Stop Word Removal on Document Similarity for Hindi Text. Research Cell: An International Journal of Engineering Sciences. 2014 Dec; 2:161-63.
39. The EMILLE Corpus.
40. Hindi WordNet. Date Accessed: 07/08/2013: Available from: http://www.cfilt.iitb.ac.in/wordnet/webhwn/index.php.
41. Similarity of Documents Based on Vector Space Model. Date Accessed: 04/03/2013: Available from: http://www.slideshare.net/dalal404/document-similarity-with-vector-space-model.