




Charismatic Document Clustering Through Novel K-Means Non-negative Matrix Factorization (KNMF) Algorithm Using Key Phrase Extraction

E. Laxmi Lydia¹  · P. Krishna Kumar² · K. Shankar³ · S. K. Lakshmanaprabu⁴ · R. M. Vidhyavathi⁵ · Andino Maseleno⁶

Received: 13 May 2018 / Accepted: 30 July 2018 / Published online: 7 August 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract

The tedious challenging of Big Data is to store and retrieve of required data from the search engines. *Problem Defined* There is an obligation for the quick and efficient retrieval of useful information for the many organizations. The elementary idea is to arrange these computing files of organization into individual folders in an hierarchical order of folders. Manually, to order these files into folders, there is an ardent need to know about the file contents and name of the files to give impression of files, so that it provides an alignment of certain set of files as a bunch. *Problem Statement* Manual grouping of files has its own complications, for example when these files are in numerous amounts and also their contents cannot be illustrious by their labels. Therefore, it's an intense requirement for Document clustering with data processing machines for enthusiastic results. *Existing System* A couple of analyzers are impending with dynamic algorithms and comprehensive analogy of extant algorithms, but, yet, these have been restricted to organizations and colleges. After recent updated rules of NMF their raised a self interest in document clustering. These rules gave trust in its performances with better results when compared to Latent Semantic Indexing with Singular Value Decomposition. *Proposed System* A new working miniature called Novel K-means Non-Negative Matrix Factorization (KNMF) is implemented using renovated guidelines of NMF which has been diagnosed for clustering documents consequently. A new data set called Newsgroup20 is considered for the exploratory purpose. Removal of common clutter/stop words using keywords from Key Phrase Extraction Algorithm and a new proposed Iterated Lovin stemming will be utilized in preprocessing step in assisting to KNMF. Compared to the Porter stemmer and Lovins stemmer algorithms, Iterative Lovins algorithm is providing 5% more reduction. 60% of the document terms are been minimized to root as remaining terms are already root words. Eventually, an appeal to these processes named "Progressive Text min-

✉ E. Laxmi Lydia
elaxmi2002@yahoo.com

Extended author information available on the last page of the article

ing radical” is developed in lateral exertion of K-Means algorithm from the defined Apache Mahout Project which is used to analyze the performance of the MapReduce framework in Hadoop.

Keywords Document clustering · K-means non-negative matrix factorization (KNMF) · Iterated lovin stemmers · Key phrase extraction · Stopwords · MapReduce · Hadoop · Term frequency-inverse term frequency (Tf-IDF)

1 Introduction

A standout amongst the most critical assignments in information and learning revelation is Data Clustering. As per Jain’s definition, “The objective of Data Clustering, otherwise called cluster analysis, is to find the common grouping(s) of an arrangement of examples, focuses, or questions” [1]. Data clustering has different applications in various fields. For instance, in Computer Vision, Image Segmentation can be characterized as a clustering issue [2]. In Information Retrieval, report grouping can give various levelled recovery and upgrades in level recovery execution [3]. In Bio informatics, grouping is utilized for enhancing different arrangement [4]. Numerous other essential applications likewise exist in fields like: Medicine, Online Social Networks, and so on [1].

Document grouping strategies have been getting an ever increasing amount attentions concerning illustration an essential and empowering device to productive organization, navigation, retrieval, and outline for colossal volumes from claiming content documents. Using great document clustering methods, computers could naturally arrange a record corpus under worthwhile group echelons, that could empower a productive scanning and route of the corpus. Next to the maintainance of corpus, appropriate best stemming algorithm is applied. The various stemming algorithms are Iterative Lovins stemmer, Lovins stemmer, Porters stemmer. It aims to minimize the words to its root. A productive document scanning furthermore route will be a profitable supplement of the deficiencies of conventional IR innovations.

Further more this paper describes the performance of document clustering focusing on the stemmer algorithms. Related work describes the existing algorithms for document clustering stemming algorithms followed by the existing problem while using LSI (Latent Semantic Indexing), SVD (Singular Value Decomposition), PCA (Principal Components Analysis). Proposed work specifies the detailed description of the document clustering and stemmer algorithms. Finally, result analysis is performed by Newsgroup20 dataset by comparing three stemmer algorithms and ICF, WSF, CSWF Factors for stemmed words.

2 Related Work

The action of compassionating the likeliness and unlikeliness across the present phenomenon’s and thus, isolating them into consequential subgroups sharing common characteristic is known as Clustering. If suppose the classification is done with abso-

lute items of relevant features, this is a sign of best clustering method adopted. In clustering we can't predict the assertive groups as this is an autonomous method. Simply 'Document Clustering' is a similar gathering of files. According to Guduru [5], Conventional techniques in document clustering utilize set of words as proportionate to discover similitude across documents. These words are thought to be commonly autonomous which in genuine application may not be the situation. Conventional VSI utilizes words to depict the documents however as a general rule the ideas/semantics/highlights/points are what portrays the documents. The extracted highlights hold the most vital thought/idea relating to the documents. Include extraction has been effectively utilized as a part of text mining with unpredictable algorithms like Principal Components Analysis (PCA), Singular Value Decomposition (SVD), and Non-Negative Matrix Factorization (NMF) including factorization of the document word matrix. In Proceedings of Berry et al. [6], Landauer et al. [7], it was said that a novel Information Retrieval technique called Latent Semantic Indexing (LSI) was sketched to call the flaws of the classic VSM model. In order to rectify the faults raised in lexical matching, LSI uses demographically derived concepts instead of isolated word retrieval. It considers some latent structures for word usages which were partly obscured by variability of word choice. To appraise the structural word usage across documents and retrieval performance with database of singular value vectors, a truncated Singular Value Decomposition (SVD) is used. Performance data shows that these statistically derived vectors are more robust indicators than individual terms. Application of Latent Semantic Indexing with results can be found in [6, 7]. Singular Value Decomposition is extensively used in standard factorization of the data matrix. As SVD vectors contain negative values compared to VSM vectors which contain positive values, it became difficult for interpretation. These issues have been replaced with an outstanding algorithm NMF (formulated in Lee and Seung [8, 9]) that have various beneficiaries over Standard PCA/SVD like non-negativity in NMF ensures coherent parts of original data (text, images etc.) [10]. Ding et al. [10] demonstrated that when Frobenius standard is utilized as a difference and including an orthogonality imperative $H^T H = I$, NMF is comparable to a casual type of K-Means clustering. Xu et al. were the primary ones to utilize NMF for document clustering in [11] where unit Euclidean separation imperative was added to column vectors in Yang et al. [12] broadened this progress by including the sparsity constraints since inadequacy is the critical feature of tremendous data in semantic space. In both works, the clustering was found to understand the components of the matrices. As per Xu et al. [11], Interpreting of two positive matrices U and V has been analogue with SVD. As per it, every element u_{ij} of matrix U and v_{ij} of matrix V represents the degree to which term $f_i \in W$ belongs to cluster j and degree document i is associated with cluster j respectively. If document i solely belongs to cluster x , then v_{ix} will take on a large value while rest of the elements in i th row vector of V will take on a small value close to zero. From the work of Kanjani [13] it is seen that the accuracy of algorithm from Lee and Seung [9] is higher than their derivatives [11, 12]. In this work, he undertook the authenticated multiplicative update proposed by Lee and Seung [9]. Porter [14] represented an elementary algorithm for stemming words of English language that has been widely adopted with extension for standard approach of word conflation for information retrieval.

The Lovins stemming algorithm proposed by Lovins [15] is mainly used in analysis of English language with reference to porter stemmer algorithm. This algorithm exhibits its functionalities in familiar paths along with its shades it appear. Lovins gave scope to porter stemmer to shape its algorithm with ample modifications and progressiveness. According to Laxmi [16], K-means have resolved the well-known clustering problem that is an unsupervised learning algorithm. This is done by assuming the ‘K’ number of clusters for classifying a given grid processor in a clear snapshot way. As the performance result analysis depends on initial centroids, K-Means clustering does not have a guarantee for an optimal solution. Thus, the proposed system uses the partition clustering (K-Centroids clustering). With the continuous work of Laxmi [16] Disparateness cluster environment is created along with the properties of resource such as resource type, processing speed, and the memory. In order to avoid the scheduling delay, the system needs to form a cluster using the K-centroids clustering. Depending up on higher priorities, the node will move to the cluster [17]. Clustering of Documents and detail description of KNMF is studied with parallel explanation of indexing the documents [18].

3 Problem Statement

In this gigantic world, we are overloaded with numerous numbers of files or documents in each of its related fields. These increase overload to the users to analyze his/her strategies for a particular group. Even though they all belong to same fields, there are various sub groups present. So to distribute these files into sub groups we need to know the content and then separate them into groups. For this we can manually work for the separation of 10–100 in number files. But if these are in huge amount, manual distribution is not enough. So computer aided work is to be acquired by sub group the files based on its file name along with its content.

Many of great scholars have done a research work in text mining related to document clustering and came out with efficient outputs as years by accordingly. Here of these works, document clustering with updated rules of NMF which was proposed by Lee and Sung gave good performance. Before this, Latent Semantic Indexing (LSI), Singular Value Decomposition (SVD), Principal Components Analysis (PCA) were in great use. Of these, LSI with the help of truncated SVD has estimated the word structures. As the ages pass by new innovations are replacing the old one with extra features which took place even in document clustering problem. Here this problem has recovered with Lee and Sung’s updated NMF rule which have better performance than LSI. But these rules are limited to academics.

4 Proposed Work

A new updated model i.e., KNMF compared to Lee and Sung’s NMFis used for Automatic Document clustering. For its experimental implementation, a New Group 20 data set is used. This model is more efficient when compared to that of NMF

proposed by Lee and Sung. As the k-means factor is added to NMF it gives a prominent importance in clustering with extracted features.

Extracted features play a dominant role in clustering of documents. These features are extracted based on the requirements we define before classification. These can be semantics, topics, and featured words from the defined document. Mainly these selected features condense the size of the documents with emphatic words. This makes the clustering amiable. To achieve this we need to gear up with sequential steps. Initially, Stopwords/Common clutters are removed with the help of keywords from Key Phrase Extraction Algorithm and Stemming from proposed Iterated Lovin Stemmer Algorithm. Later, Performance of MapReduce framework in Hadoop is achieved through the Parallel implementation of K-Means clustering algorithm.

Here in this paper, the first portion of implementation is elaborated that include to free the required documents from unwanted clutters and evaluate its TF-IDF (Term frequency-Inverse Document frequency) count values for clustering process along with novel stemming algorithm.

In the proposed system of KNMF, the clustering of documents is curved between similarities of isolated documents and defined characteristics. If suppose in the computation of KNMF, these characteristics indulge basic vectors like $W = \{w_1, w_2, w_3, \dots, w_x\}$ and Term Document matrix related documents like $V = \{d_1, d_2, \dots, d_i\}$ then document d_i would replace with vector w_x , with condition that angle between d_i and w_x is less.

5 The Methodology Adapted

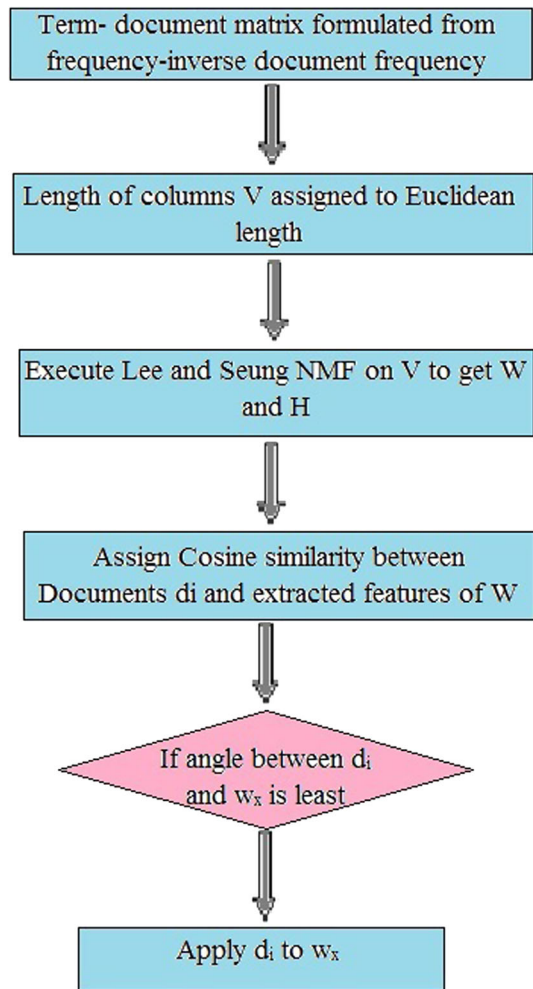
1. Formulate Term Document Matrix V using Term frequency –Inverse document frequency for those groups of file folders.
2. Euclidean length is formalized by the column length of V .
3. We formulate W and H by calibration of Lee and Seung NMF and KNMF.
4. Cosine Similarity is administered to calculate distance across document d_i and defined characteristics W .
5. Similar to K-means single turn algorithm, we empower d_i to w_x as the angle between them is less (Fig. 1).

Hadoop has initialized in both the Local Reference Mode and the Pseudo-Distributed Mode for running parallel versions of K-Means by submitting to the Job-Client. The time analysis measures are updated for future reference.

5.1 Steps for Initial Progressive Updating of the Documents in a Folder

1. Adjudicate the Document is novel or not. If its new, then update in index
2. Now create a Text Document if it's a new document updated in index by replacing the old one.
3. Using the Key Phrase Extraction algorithm defining 499 stopwords, stopwords in the documents are extracted. These stopwords are maintained as a text document which makes easy for modification.

Fig. 1 Flowchart of the proposed methodology



4. Now the stopwords are read and removed from the document. User can also add words to these stopwords text document.
5. Stemming algorithm is applied on the given document.
6. Created Text Document is stored in index.
7. Lastly stray files(doc that are eliminated from set of group but are found to be in)removed.

The above Fig. 2 represents the flow diagram for the detailed description of module 1 to be presented in this paper from the whole process. The module 2 describes calculation of Term Frequency-Inverse Document Frequency (TF-IDF) Value for the given input documents. Here is a block diagram of the prescribed process for module 1 and module 2 described in this paper (Fig. 3).

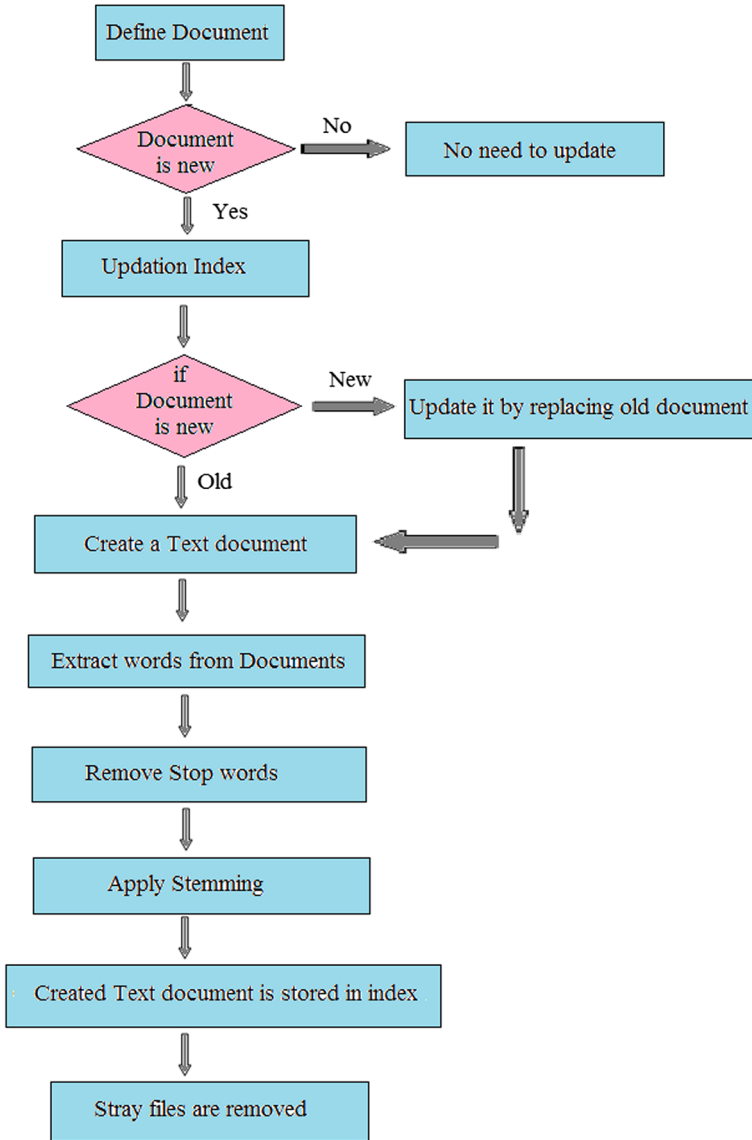


Fig. 2 Flow chart representing the Indexing of documents

5.2 Stopwords Count Algorithm

Step1: The text document to be applied is isolated into individual words and is stored in array formats.

Step2: Each isolated stopword is read from the list of stopwords defined.

Step3: Now each word is compared with the list of stopwords using sequential array technique.

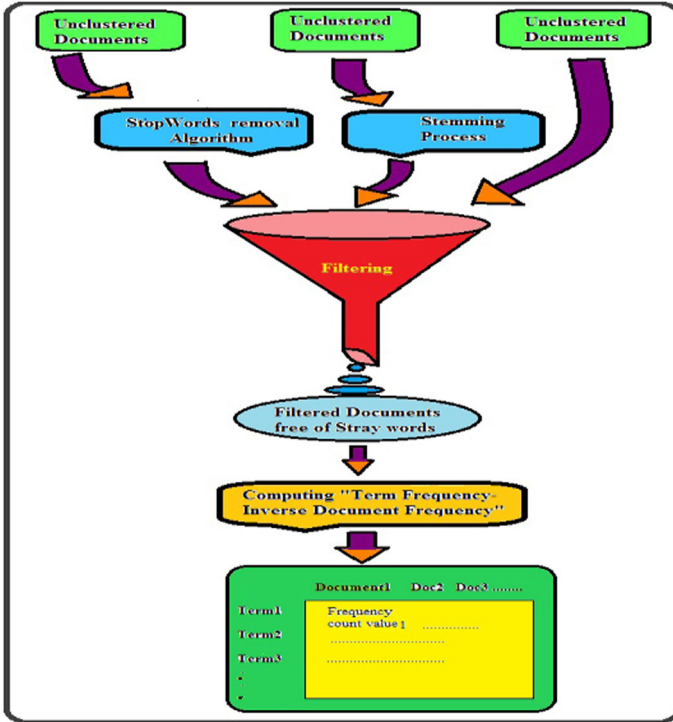


Fig. 3 Block diagram for the module 1 and 2

Step4: The word is removed from the text document if they match with words in stopwords list.

Step5: This process continues across whole text document and produces stopwords free text document.

```
//Defining stopWords list
Static Set<String>stopWordsSet = new HashSet<String>();
// adding stopwords list to s
For loop (String s: stopWords)
{
stopWordsSet.add(s);
}
// counters are defined to count stopwords
static enum Counters
{
StopWords;
}
//mapper phase for identifying stopwords in given text
Comparing StopWord with string s;
If (stopWord set contains string s) // if condition
then
```



```

return s;
assigning string value to str;
defining a new string tokenizer for str that is tokens;
whilecondition (tokens are added)
{
each token added is assigned to a string 'word' ;
ifcondition (comparing the 'word' with StopWord)
{
If true then increment COUNTER value by 1;
}}}

```

5.3 Lovins Stemmer Algorithm

The algorithm consists of two steps:

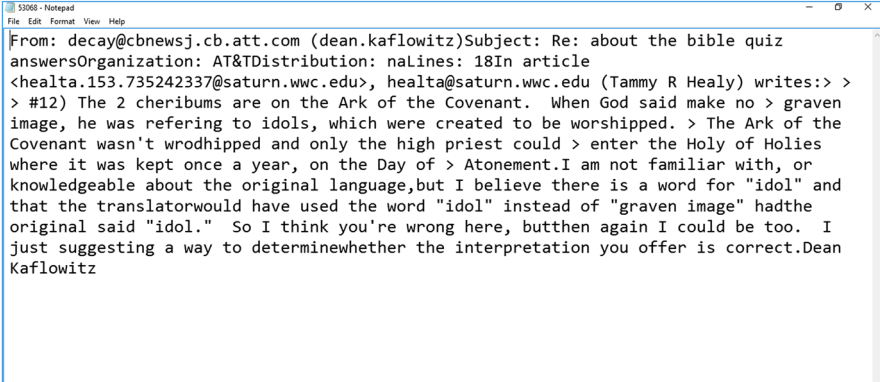
- Step 1 By eradicating the suffixes along with the treatment of held stemmers. This Suffix Stripping consists of concatenating the endings of the word with long suffix from the list of 294 suffixes and defined 29 rules.
- Step 2 Now comes the Recoding phase where these held stems are left to clear linguistic clauses similar to double letter endings of 'd' and 't'. According to the phase, it also depends on 35 rules for termination of stems. Example: 'Outputting' can be stemmed as 'Output' rather than 'Outputt'.
- Step 3 Lastly, with the help of partly similar matching algorithm, fusion is rectified. It makes an rapid increase in fusion level with illusions of having two equal stems bearing a diversities as a reason of suffix stemming methods. Example: 'EXPLAIN' and 'EXPLANATION' can be stemmed distinctly with 'EXPLAIN' and 'EXPLAN'.

5.4 Porter Stemmer Algorithm

- Step 1 This step in the algorithm plays a complicate role with 3 shares of main definition. The first share deals with plurals, for example sses -> ss and removal of s.
- Step 2 The second share avoids the ending letters like 'ED' and 'ING' and implement 'EED' applied when applicable. This share proceeds if 'ED' and 'ING' are terminated with translating left over stem to conform that particular suffices are recognized later.
- Step 3 The third share simply manipulate one of the terminal 'Y' to 'I'.
- Step 4 The remaining steps in this stemmer contain rules to deal with different order classes of suffices, initially transforming double suffices to a single suffix and then removing suffices provided the relevant conditions are met.

5.5 Proposed Iterated Lovins Stemmer Algorithm

- Step 1: Iterated stemmer is extension of lovins stemmer. So it inherits the rules of the lovins stemmer.



```

51060 - Notepad
File Edit Format View Help
From: decay@cbnewsj.cb.att.com (dean.kafLOWITZ)Subject: Re: about the bible quiz
answersOrganization: AT&TDistribution: naLines: 18In article
<healta.153.735242337@saturn.wwc.edu>, healta@saturn.wwc.edu (Tammy R Healy) writes:> >
> #12) The 2 cheribums are on the Ark of the Covenant. When God said make no > graven
image, he was refering to idols, which were created to be worshipped. > The Ark of the
Covenant wasn't wrodhipped and only the high priest could > enter the Holy of Holies
where it was kept once a year, on the Day of > Atonement.I am not familiar with, or
knowledgeable about the original language,but I believe there is a word for "idol" and
that the translatorwould have used the word "idol" instead of "graven image" hadthe
original said "idol." So I think you're wrong here, butthen again I could be too. I
just suggesting a way to determinewhether the interpretation you offer is correct.Dean
KafLOWITZ

```

Fig. 4 Sample dataset from Newsgroup20

- public class Iterated Lovins Stemmer extends Lovins Stemmer
- Step 2: Before Iterating stemmer of the given word the particular word has to be changed to lower case.
- ```

/*Defining a string STR;
Applying a 'IF' condition to check the given length of STR is greater than
1;
If so then need to convert the words into lower case by equalizing the stems.
This runs with a loop function. */

```
- Step 3: Here in this algorithm the Lovinsstemer procedure carries out in reoccurrence way (may be twice) to avoid the extreme left over stemmed words.

## 5.6 Experimental Setting and Data Description

For the experimental purpose Fig. 4 has been taken as a sample input from 20 Newsgroup.20 NewsGroup is significant relevant information set for clustering and classification. It has an accumulation around 20,000 reports cross-wise over 20 diverse Newsgroups from Usenet. Each of these Newsgroups is gathered in a sub-directory with every clause gathering in a big document. The newsgroup data set has been executed on the following software's.

|                                              |                                                                |
|----------------------------------------------|----------------------------------------------------------------|
| Software Version                             | 1.0                                                            |
| Programming Language Referred                | Java                                                           |
| Platform Used                                | Only tested in GNU/Linux                                       |
| Preferred IDE                                | NetBeans IDE 6.0.1                                             |
| Apache Cloudera                              | 3                                                              |
| High end Server                              | PN: 7382IA4 Two socket tower Intel Xeon ES<br>2403(Quard core) |
| Cluster with Hadoop software having 15 nodes |                                                                |

```

cloudera-demo-03.7 - VMware Workstation 12 Player (Non-commercial use only)
Player
Applications Places System
cloudera@cloudera-vm: ~
File Edit View Search Terminal Help
cloudera@cloudera-vm:~$ hadoop fs -put 53068.txt /home/cloudera
put: Target /home/cloudera already exists
cloudera@cloudera-vm:~$ hadoop fs -put 53068.txt /user/cloudera
cloudera@cloudera-vm:~$ ls
53068.txt om StopWords.java WordCount.class
alphabet.java Stemmer.class StopWords$StopWordsMapper.class wordcount.jar
cloudera stemming3.java test WordCount.java
Code StopList TfIdf1.class WordCounts$WordCountMapper.class
commons-cli-1.2.jar StopList.txt TfIdf1.java WordCounts$WordCountReducer.class
Desktop stopword.jar TfIdf.class wp.txt
hadoop-core-0.20.2-cdh3u0.jar StopWords.class _user_cloudera_om
Key-valueDocs StopWords$SCOUNTERS.class _user_cloudera_wp.txt
cloudera@cloudera-vm:~$ export CLASSPATH=${HADOOP_HOME}/hadoop-core-0.20.2-cdh3u0.jar:commons-cli-1.2.jar:${CLASSPATH}
cloudera@cloudera-vm:~$ javac StopWords.java
cloudera@cloudera-vm:~$ ls
53068.txt om StopWords.java WordCount.class
alphabet.java Stemmer.class StopWords$StopWordsMapper.class wordcount.jar
cloudera stemming3.java test WordCount.java
Code StopList TfIdf1.class WordCounts$WordCountMapper.class
commons-cli-1.2.jar StopList.txt TfIdf1.java WordCounts$WordCountReducer.class
Desktop stopword.jar TfIdf.class wp.txt
hadoop-core-0.20.2-cdh3u0.jar StopWords.class _user_cloudera_om
Key-valueDocs StopWords$SCOUNTERS.class _user_cloudera_wp.txt
cloudera@cloudera-vm:~$ jar cvf stopword.jar StopWords*.class
added manifest
adding: StopWords.class(in = 7990) (out= 4499)(deflated 43%)
adding: StopWords$SCOUNTERS.class(in = 848) (out= 479)(deflated 43%)
adding: StopWords$StopWordsMapper.class(in = 1897) (out= 853)(deflated 55%)
cloudera@cloudera-vm:~$ hadoop stopword.jar StopWords 53068.txt stopword.txt

```

Fig. 5 Adding the input file to HDFS and creating a Stop-words jar file

Since Hadoop is worked with java, it is simple for interoperability. The below screens shoots has been generated after executing in the Big data Analytics cluster which was created as phase 1 of the project. Figure 4 screen shot shows the piece of Input data taken from newsgroup 20. Figure 5 Screen shot shows the adding of the given input file into HDFS. It also shows execution of StopWords program by creating a stopwords.jar file along with some paths needed for the compiling. Figure 6 shows the output screen for compiling of the Stopwords program with the required stopwords count in a given input file taken from the newsgroup20 dataset. Figure 7 screen shot shows the proposed Iterated Lovins Stemming Output of the Fig. 4 input resulting with maximum minimized stem words. Figure 8 Screen shot shows the Lovins Stemming Output of the Fig. 4 input. The Fig. 9 Screen shot shows the Porter Stemming Output of the Fig. 4 input. Figure 10 Screen shot shows the Tf-Idf count values of the sample input data files taken.

## 5.7 Screen Shoots Through Experimental Setting

## 6 Result Analysis

Here this section elevates the performance strategies of different stemming algorithms: Iterated Lovins Stemming Algorithm, Lovins Algorithm, and Porter Stemming Algorithm. The analysis metrics considered here are: Index Compression Factor (ICF), Word Stemming Factor (WSF), Correct Stemming Word Factor (CSWF).

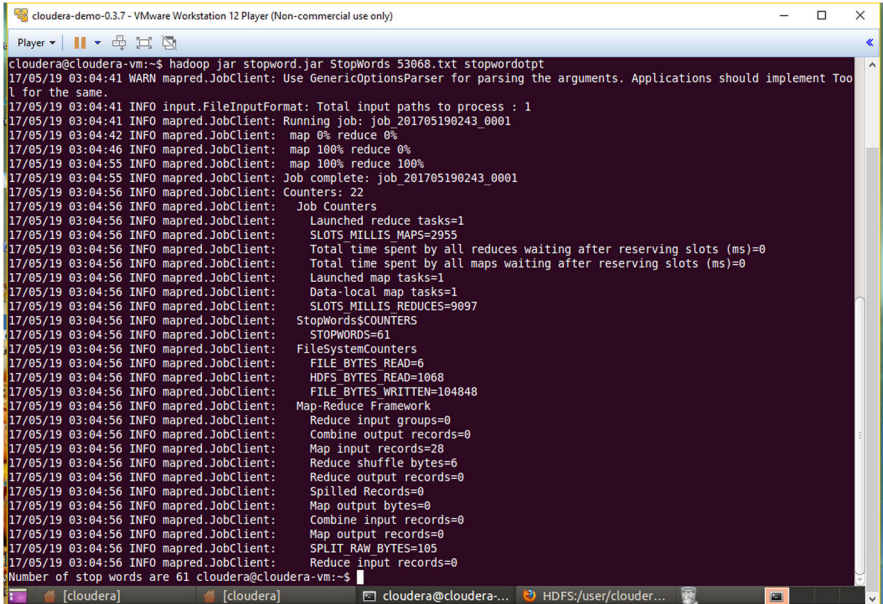


Fig. 6 Output screen showing the count of stop words in a given input file through Map Reduce phase

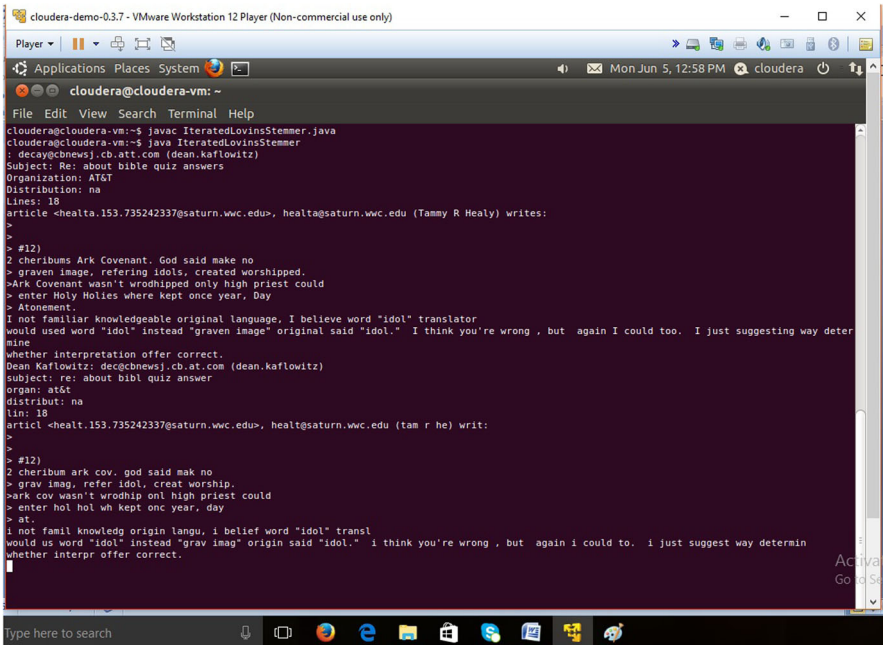


Fig. 7 Output screen for proposed iterated Lovins stemming algorithm

```

cloudera@cloudera-vm:~$ javac LovinsStemmer.java
Note: LovinsStemmer.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
cloudera@cloudera-vm:~$ java LovinsStemmer
; deca@cbnewsj.cb.att.com (dean.kaflowitz)
Subject: Re: about bible quiz answers
Organization: AT&T
Distribution: na
Lines: 18
article <healta.153.735242337@saturn.wcc.edu>, healta@saturn.wcc.edu (Tammy R He
aly) writes:
>
> #12)
2 cherubms Ark Covenant. God said make no
> graven image, refering idols, created worshipped.
>Ark Covenant wasn't wrodhipped only high priest could
> enter Holy Holies where kept once year, Day
> Atonement.
I not familiar knowledgeable original language. I believe word "idol" translator
would used word "idol" instead "graven image" original said "idol." I think you
're wrong , but again I could too. I just suggesting way determine
whether interpretation offer correct.
Dean Kaflowitz: deca@cbnewsj.cb.att.com (dean.kaflowitz)
subject: re: about bibl quiz answer
organ: at&t
distribut: na
lin: 18
articl <healt.153.735242337@saturn.wcc.edu>, healt@saturn.wcc.edu (tam r hea) wr
it:
>
> #12)
2 cheribum ark coven. god said mak no
> grav imag, refer idols, creat worship.
>ark coven wasn't wrodhip onl high priest could
> enter hol hol whes kept once year, day
> aton.
i not famili knowledge origin langu, i belief word "idol" transl
would us word "idol" instead "grav imag" origin said "idol." i think you're wro
ng , but again i could to. i just suggest way determin
whether interpres offer correct.

```

Fig. 8 Output screen for Lovins stemming algorithm

## 6.1 Index Compression Factor (ICF)

It defines the percentage of the total number of apparent words to that of Number of apparent stems after stemming. The strength of the stemmer increases along with this ICF Value.

$$ICF = \frac{(N - S)}{S} \times 100$$

The Fig. 11 defines the performance metric Index Compression Factor (ICF) considering three documents doc 1, doc 2, doc 3 performing comparison on three stemmer algorithms Iterated Lovins algorithm, Lovins algorithm and Porters algorithm. Here the graph shows that Iterated algorithm performs best than the other two.

## 6.2 Word Stemming Factor (WSF)

It is defined as percentage of words that have been stemmed by the stemming process out of the total words in a sample. Strength of Stemming increases along with number of words stemmed.

$$WSF = \frac{WS}{TW} \times 100$$

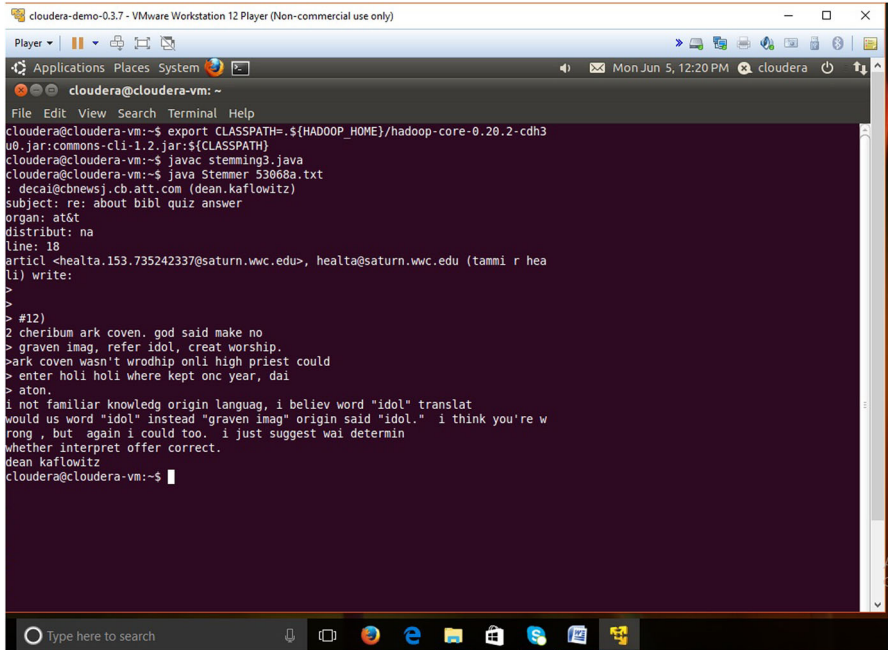


Fig. 9 Output screen for Porter stemming algorithm

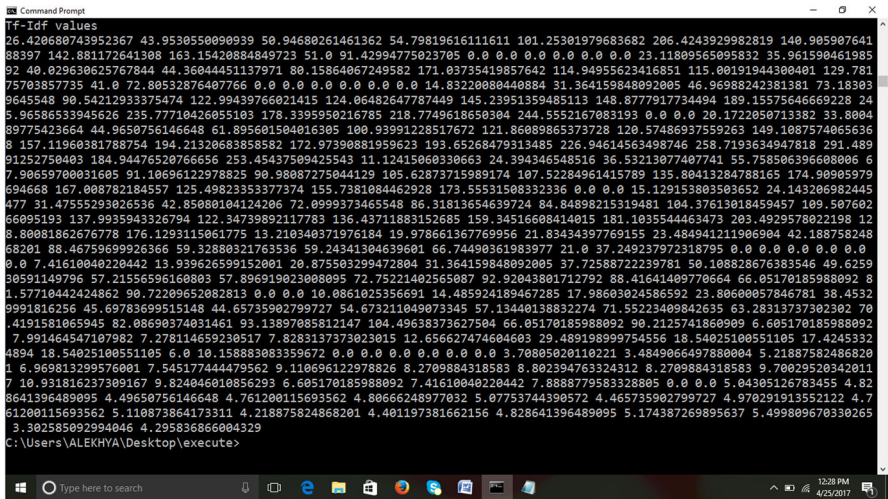


Fig. 10 Output screen showing Tf-Idf values of sample input files taken

The Fig. 12 defines the performance metric Word Stemming Factor considering three documents performing comparison on three stemming algorithms. Here the graph shows that Iterated Lovins stemmer algorithms performs efficiently than others.



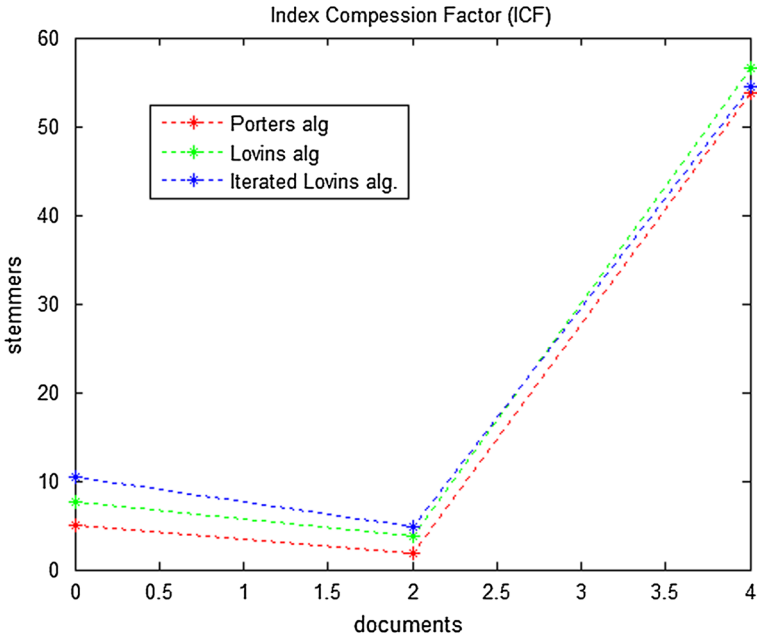


Fig. 11 Graph showing the ICF value result analysis of Stemmers

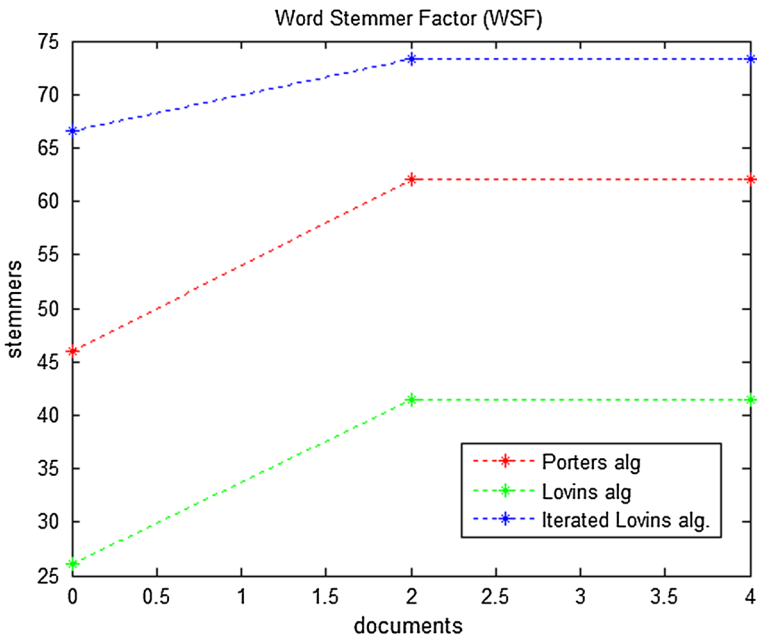


Fig. 12 Graph analysis of WSF values of Stemmers

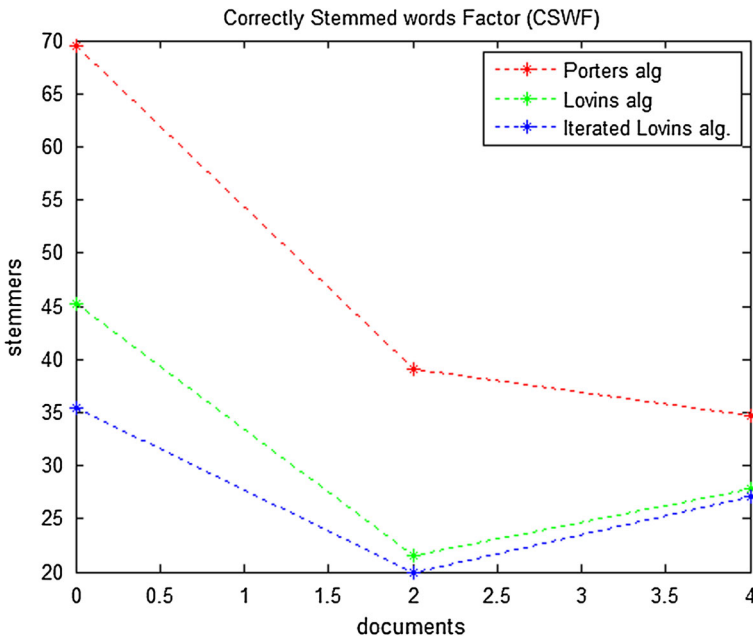


Fig. 13 Graph showing the stemming words versus distinct words in each doc for stemmers

### 6.3 Correct Stemming Word Factor (CSWF)

It is defined as percentage of words that have been stemmed correctly out of the number of words stemmed. The accuracy of the stemmer increases with increased percentage of CSWF.

$$CSWF = \frac{CSW}{WS} \times 100$$

The Fig. 13 defines the performance metric Correct stemming word factor providing accurate stem words by considering three documents and performing comparison of three algorithms, among which iterative algorithm achieves the correct word most among three algorithms.

The following Table 1 shows the result analysis of different Stemmers {Iterated Lovins Stemmer, Lovins Stemmer, and Porter Stemmer} considering 3 documents {Doc1 as 'D 1', Doc2 as 'D 2', and Doc3 as 'D 3'} as examples.

The above Table 1 and graph represents result analysis of different stemmers (Iterated Lovins, Lovins, Porter). The table shows the factors that considered for efficient stemmer analysis. These factors help us to select the best stemming algorithm that suits our project. These values are calculated based on the formulas mentioned above.

Result analysis with the stopword algorithm and stemming algorithm:

- Size of the data is reduced.
- Execution time reduced as file size is less.



**Table 1** Stemming factors defined for various stemmers considering 3 documents

| Stemmer analysis                            | Iterated Lovins Stemmer |       |       | Lovins Stemmer |       |       | Porter Stemmer |       |       |
|---------------------------------------------|-------------------------|-------|-------|----------------|-------|-------|----------------|-------|-------|
|                                             | D 1                     | D 2   | D 3   | D 1            | D 2   | D 3   | D 1            | D 2   | D 3   |
| Documents                                   |                         |       |       |                |       |       |                |       |       |
| Total Words (TW)                            | 50                      | 157   | 1858  | 50             | 157   | 1858  | 50             | 157   | 1858  |
| Number of unique words before stemming (N)  | 42                      | 109   | 1571  | 42             | 109   | 1571  | 42             | 109   | 1571  |
| Number of Distinct words after stemming (S) | 38                      | 104   | 680   | 39             | 105   | 683   | 40             | 107   | 727   |
| Index Compression Factor (ICF)              | 10.52                   | 4.80  | 54.43 | 7.69           | 3.80  | 56.52 | 5              | 1.86  | 53.72 |
| Number of words Stemmed (WS)                | 31                      | 65    | 1363  | 31             | 65    | 1363  | 23             | 41    | 1237  |
| Words Stemmed Factor (WSF)                  | 62                      | 41.40 | 73.35 | 62             | 41.40 | 73.35 | 46             | 26.11 | 66.58 |
| Number of correctly stemmed words (CSW)     | 11                      | 13    | 369   | 14             | 14    | 379   | 16             | 16    | 430   |
| Correctly Stemmed Words Factor (CSWF)       | 35.48                   | 20    | 27.07 | 45.16          | 21.53 | 27.80 | 69.56          | 39.02 | 34.76 |

- Success rate obtained very quickly.
- Clustering can be easily applied.
- Less time consumption.
- Preserves system energy with less time period.

## 7 Conclusion

In this historical world, we are overloaded with numerous numbers of files or documents in each of its related fields. As the generation moves the historical and future trends makes innovations leaving behind enormous amount of data. So to process and analyze this huge data “Big Data” approaches gave relief from the database management tools and traditional data processing applications. A comparison is performed among Iterated Lovins algorithm, Lovins algorithm and Porters algorithms with comparative factors using ICF, WSF, CSWF resulting maximum minimized stem words by Iterated Lovins algorithms.

Thus a new algorithm KNMF is used and the application will be named as ‘Progressive Text Mining radical’. Therefore with those defined characteristics of KNMF help to cluster the documents as we consider them to be ultimate labels of clusters in k-means. And also parallel implementation with MapReduce for huge sized documents lead to minimize time computation.

**Acknowledgement** This work is financially supported by the Department of Science and Technology (DST), Science and Engineering Research Board (SERB) under the scheme of ECR. We thank DST\_SERB for the financial support to carry the research work.


## References

1. Jain, A.K.: Data clustering: 50 years beyond K-means. *Pattern Recognit. Lett.* **31**(8), 651–666 (2010)
2. Zhang, H., Fritts, J.E., Goldman, S.A.: Image segmentation evaluation: a survey of unsupervised methods. *Comput. Vis. Image Underst.* **110**(2), 260–280 (2008)
3. Baeza Yates, R., Ribeiro Neto, B., et al.: *Modern Information Retrieval*. ACM Press, New York (1999)
4. Miller, D.J., Wang, Y., Kesidis, G.: Emergent unsupervised clustering paradigms with potential application to bioinformatics. *Front. Biosci.* **13**(1), 677–690 (2008)
5. Guduru, N.: *Text Mining with Support Vector Machines (SVM) and Non-Negative Matrix Factorization (NMF) Algorithm*. Master’s Thesis, University of Rhode Island, CS Department (2006)
6. Berry, M.W., Dumais, S.T., O’Brien, G.W.: Using linear algebra for intelligent information retrieval. *SIAM Rev.* **37**(4), 573–595 (1994)
7. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. *Discourse Process.* **25**(2–3), 259–284 (1998)
8. Lee, D.D., Seung, H.: Learning the parts of objects by non-negative matrix factorization (NMF). *Nature* **401**, 788–791 (1999)
9. Lee, D.D., Seung, H.: Algorithm for non-negative matrix factorization. In: Dietterich, T.G., Tresp, V. (eds.) *Advances in Neural Information Processing Systems*, Volume 13, Proceedings of the Conference: 556562. The MIT Press
10. Ding, C., He, X., Simon, H.D.: On the equivalence of nonnegative matrix factorization (NMF) and spectral clustering. In: *Proceedings of the 2005 SIAM International Conference on Data Mining*, pp. 606–610. Society for Industrial and Applied Mathematics (2005)
11. Xu, W., Liu, X., Gong, Y.: Document clustering based on NON-negative matrix factorization. In: *Proceedings in ACM SIGIR*, pp. 267–273 (2003)

12. Yang, C.F., Ye, M., Zhao, J.: Document clustering based on non-negative sparse matrix factorization. In: International Conference on Advances in Natural Computation, pp. 557–563 (2005)
13. Kanjani, K.: Parallel Non Negative Matrix Factorization for Document Clustering. CPSC-659 (Parallel and Distributed Numerical Algorithms) Course. Texas A&M University, Tech. Rep. (2007)
14. Porter, M.F.: An algorithm for suffix stripping. Program **14**(3), 130–137 (1980)
15. Lovins, J.B.: Development of a stemming algorithm. Mech. Translat. Comp. Linguistics **11**(1–2), 22–31 (1968)
16. Laxmi, H.V.T.E.V., Somasundaram, K.: 2HARS: heterogeneity-aware resource scheduling in grid environment using K-centroids clustering and PSO techniques. Int. J. Appl. Eng. Res. **10**(7), 18047–18062 (2015)
17. Laxmi Lydia, E., Ben Swarup, M., Narsimham, C.: A disparateness-aware scheduling using K-centroids clustering and PSO techniques in hadoop cluster. Int. J. Adv. Found. Res. Comput. **2**(12) (2015)
18. Laxmi Lydia, E.: Text mining with lucene and hadoop: document clustering with updated rules of NMF non-negative matrix factorization. Int. J. Pure Appl. Math. **118**, 191–198 (2018)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

## Affiliations

**E. Laxmi Lydia**<sup>1</sup>  · **P. Krishna Kumar**<sup>2</sup> · **K. Shankar**<sup>3</sup> · **S. K. Lakshmanaprabu**<sup>4</sup> · **R. M. Vidhyavathi**<sup>5</sup> · **Andino Maseleno**<sup>6</sup>

P. Krishna Kumar  
ponkrishkumar@yahoo.com

K. Shankar  
shankarcrypto@gmail.com

S. K. Lakshmanaprabu  
prabusk.leo@gmail.com

R. M. Vidhyavathi  
vidhyamiss@gmail.com

Andino Maseleno  
andimaseleno@gmail.com

- <sup>1</sup> Department of Computer Science Engineering, Vignan's Institute of Information Technology, Duvvada, Andhra Pradesh, India
- <sup>2</sup> Department of Computer Science and Engineering, V V College of Engineering, Tuticorin District, Tamil Nadu, India
- <sup>3</sup> School of Computing, Kalasalingam Academy of Research and Education, Krishnankoil, India
- <sup>4</sup> Department of Electronics and Instrumentation Engineering, BS Abdur Rahman Crescent Institute of Science and Technology, Chennai, India
- <sup>5</sup> Department of Bioinformatics, Alagappa University, Karaikudi, India
- <sup>6</sup> Department of Informatics Management, STMIK Pringsewu, Pringsewu, Lampung, Indonesia