ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 4, June 2012

Auto-assemblage for Suffix Tree Clustering

Pushplata, Mr Ram Chatterjee

***Abstract*— Due to explosive growth of extracting the information from large repository of data, to get effective results, clustering is used. Clustering makes the searching efficient for better search results. Clustering is the process of grouping of similar type content. Document Clustering; organize the documents of similar type contents into groups. Partitioned and Hierarchical clustering algorithms are mainly used for clustering the documents. In this paper, k-means describe the partitioned clustering algorithm and further hierarchical clustering defines the Agglomerative hierarchical clustering and Divisive hierarchical clustering. The paper presents the tool, which describe the algorithmic steps that are used in Suffix Tree Clustering (STC) algorithm for clustering the documents. STC is a search result clustering, which perform the clustering on the dataset. Dataset is the collection of the text documents. The paper focuses on the steps for document clustering by using the Suffix Tree Clustering Algorithm. The algorithm steps are display by the screen shots that is taken from the running tool.**

***Keywords*— Data Mining, Document Clustering, Hierarchical Clustering, Information Retrieval, Partitioned Clustering, Score Function, Similarity Measures, Suffix Tree Clustering, Suffix Tree Data model, Term Frequency and Inverse Document Frequency.**

I. INTRODUCTION

Due to explosive growth of availability of large volume of data electronically, that creates a need to automatically explore the large data collections. Clustering [14] algorithms are unsupervised, fast and scalable. Clustering is the process of dividing the set of objects into specific number of clusters. Document clustering arises from information retrieval domains, "It finds grouping for a set of documents belonging to the same cluster are similar and documents belongs to the different cluster are dissimilar". Information retrieval plays an important role in data mining for extracting the relevant information for related to user request.

Information retrieval finds the file contents and identifies their similarity. It measures the performance of the documents by using the precision and recall. Cluster

Analysis organize the data by abstracting underlying structure either as a grouping of individuals or as a hierarchy of groups. The representation can be done if the data groups according to preconceived ideas. The paper also presents the document clustering and a small introductory part about the partitioned and hierarchical document clustering techniques. The main focus of the paper is implementing the steps of Suffix Tree Clustering algorithm for information retrieval. The tool "Auto Assemblage Version 1.0.0" defines the algorithmic steps of the Suffix Tree Clustering [13] on the text documents. The tool also defines the Suffix Tree Clustering with Binary Similarity and Cosine Similarity measures for clustering the documents.

Data mining [2] tools predict the future behavior and trends. There are two different clustering algorithms are used in the paper i.e. partitioned clustering and hierarchical clustering. Partitioned clustering techniques are well suited for clustering the large volume of document datasets due to their low computational requirements. The time complexity is almost linear in partitioning clustering techniques. Hierarchical clustering produces the hierarchy of the documents for clustering the documents.

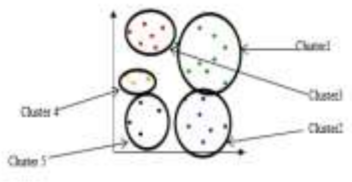


Figure 1: Cluster Example

II. RELATED WORK

Document clustering [3], [14] is a technique that originates from the data mining. Data mining [2] is used to retrieve the information from the large repository of the data. It has the methods such as classification, regression, clustering and summarization. We use the term document clustering to cluster the documents for efficient search results. Document clustering is used for information retrieval to retrieve the information form the text documents. In previous work document clustering are improved according to the requirements of the users. Document clustering is used for clustering the text documents as well as the web documents. Web document clustering used for web mining. In literature survey, document clustering is the process of grouping the similar type of documents in to one clusters and dissimilar documents are in other cluster. The two of them are partitioned clustering [4] and hierarchical clustering [6]. Partitioned clustering partitioned the n data objects in to k number of clusters. These k numbers of cluster are selected randomly from the data objects. And the hierarchical clustering can be divided into agglomerative (bottom-up) and divisive hierarchical clustering (top-down). The clustering process is mainly based on the similarity measures between the documents. The documents which are more similar according to the clustering algorithm are taken into single cluster. For measuring the similarity Manhattan and Euclidean distance measures are used that is described in the below section. Suffix tree clustering is one of the hierarchical document clustering. In the previous work Zamir and Etzioni firstly introduced the Suffix Tree Clustering Algorithm. But the tool is not implemented which follow the Suffix Tree Clustering algorithm [1], [8], [9]. In the previous work similarity measures such as binary similarity measures and cosine similarity measures can be used to measuring the similarity between the data objects. the papers which is being studied describe the data mining , document clustering, document clustering algorithms, k-means clustering algorithm, Agglomerative Hierarchical clustering, Divisive Hierarchical clustering and the last one is the Suffix Tree Clustering algorithm.

STC have some advantages over the other clustering algorithm such as: there is no requirement to specify the number of clusters, shared phrases describe the resultant clusters, and single document may appear in more than one cluster. STC has readable labels and descriptive summaries for resultant clusters.

III. DOCUMENT CLUSTERING

Document clustering [3], [14] is still a developing field which is undergoing evolution. It finds the grouping for set of documents so that documents belong to the same cluster are similar and documents belong to different clusters are dissimilar. Document clustering is a method of automatically organize the large data collection into groups. These groups are known as clusters. Document clustering treated a document as a bag of words and clustering criteria is based on the presence of similar words in document. Document clustering has always been used to improve the performance of retrieval of information from large data collection. Partitioned clustering algorithms and hierarchical clustering algorithms are two main approaches that are used in this paper.

Partitioned clustering algorithms are applied on the numerical datasets. Partitioned clustering algorithm divide the N data objects into K number of clusters. K number of clusters is pre-specified and randomly selected. K-means [4], [5] clustering algorithm is an example of the partitioned clustering algorithm. The algorithm is based on the distance between the objects. The distance can be calculated by using the distance measure functions such as Manhattan Distance and Euclidean Distance measures. The formula which is used for calculating the distance is:

In **Euclidean distance, [14]** the distance can be measured between two points such as X (x1, x2) and Y (y1, y2).

$$d= \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad (1)$$

In **Manhattan distances, [14]** the distance can be measured between two pair of objects are:

$$d(i,j)=|x_{i1}-x_{j1}| + |x_{i2}-x_{j2}|+\ldots\ldots\ldots.+|x_{in}-x_{jn}| \quad (2)$$

K-means clustering produces an effective search results while producing the clusters. Many researchers would work on improving the performance of the k-means clustering. The algorithm produces the results in different clusters depending on the randomly selected initial centroid. The algorithm work in two phases: the first phase is the randomly selection of the k centers. In the next phase, each point belonging to given dataset and assign to its nearest center.

Hierarchical clustering algorithm produces a hierarchy of clusters. Hierarchical clustering does not require to pre-specifying the number of clusters. Hierarchical clustering algorithm group the data objects in to a tree of clusters. Hierarchical clustering uses the hierarchical decomposition of a given set of data objects. Hierarchical clustering comes at the cost of lower efficiency. Hierarchical clustering represents the documents in the tree structure. Hierarchical clustering can be divided into two categories that are Agglomerative (bottom-up) and Divisive (bottom-up).

Agglomerative hierarchical clustering [14] algorithm uses the bottom-up approach it treats each document as a singleton cluster and then merges them into a single cluster that contains all the documents. The groups can be merged according the distance measures. The merging is stopped when all the objects are into a single group.

The hierarchical clustering [6], [7], [14] can be represented by Dendrogram; it is a tree like structure that shows the relationship between the objects. Dendrogram represent the each merge by the horizontal line. The similarity measures can be calculated in Agglomerative hierarchical clustering by using the methods known as:

- Single-Linkage clustering
- Complete-Linkage clustering
- Group-average clustering

In *Single-Linkage clustering* calculates the similarity between two clusters based on most similar members of the cluster. In this clustering, the minimum distance is calculated between the documents.

$$\min\{d(x,y) : x \in \mathcal{A}, y \in \mathcal{B}\}. \quad (3)$$

In *Complete-Linkage clustering* calculate the similarity of their most dissimilar members the maximum distance is calculated between the documents.

$$\max\{d(x,y) : x \in \mathcal{A}, y \in \mathcal{B}\}. \quad (4)$$

In *Group-average clustering* evaluates the cluster quality based on all similarity between the documents. The mean distance is calculated between the documents.

$$\frac{1}{|\mathcal{A}| \cdot |\mathcal{B}|} \sum_{x \in \mathcal{A}} \sum_{y \in \mathcal{B}} d(x,y). \quad (5)$$

Divisive hierarchical clustering algorithm is performing the reverse functionality as compare to the Agglomerative hierarchical clustering approach. It starts from the one group of all the objects and successively split the group into smaller ones, until each object fall in one cluster. Divisive approach divide the data objects into disjoint groups in every step and follow the same pattern until all objects fall into a separate cluster.

There is another type of hierarchical clustering algorithm which is the base of the paper. The paper is based on the tool “Auto Assemblage version 1.0.0”, which performs the algorithmic steps of Suffix Tree Clustering (STC) algorithm. The Suffix Tree Clustering Algorithm described in details in the next section.

IV. SUFFIX TREE CLUSTERING ALGORITHM

There is another type of document clustering, which is known as suffix Tree Clustering [1] (STC). The suffix tree clustering is used for improving the searching speed while performing the searching. It is a search result clustering technique to perform the searching which makes the searching efficient. The tool which is implemented performs the clustering on the text documents.

Suffix tree clustering [8], [9], [10], [12], [13] is a hierarchical document clustering, which is used for extracting the information from large repository of the data. The data which is being used in the tool for clustering the documents is the collection of the text datasets. Text datasets is the collection of the text documents. The suffix tree clustering uses the phrases (sequence of words) for clustering the documents. The Suffix Tree Clustering uses the suffix data structure for clustering the documents. It uses a tree structure for shared suffixes of the documents. Suffix tree clustering produces the results according to user query. It is a linear time clustering algorithm that means the documents are linear in size. The simplest form of the suffix tree clustering is the phrase based clustering. The example of the suffix tree structure is:

A suffix tree [11] is a data structure that allow many problems on strings (sequence of characters) to be solved quickly.

String = ‘mississippi’
Substring = ’issi’
Prefix = ’miss’
Suffix = ’ippi’

T1 =’mississippi’
T2 =’ississppi’
T3 =’ssissippi’
T4 =’sissippi’
T5 =’issippi’
T6 =’ssippi’
T7 =’sippi’
T8 =’ippi’
T9 =’ppi’
T10 =’pi’
T11 =’i’
T12 =’’

Suffixes are sorted:

T11= ’ i ’
T8 = ’ ippi ’
T5 = ’ issippi ’
T2 = ’ ississppi ’
T1 = ’ mississippi ’

T10= ’ pi ’
T9 = ’ ppi ’
T7 = ’ sippi ’
T4 = ’ sissippi ’
T6 = ’ ssippi ’
T3 = ’ ssissippi ’

Construction of a tree-

Substrings:

```
Tree-----|----> mississippi              m : mississippi
         |----> i--> |--ssi-->|--ssippi  i   : ississippi
         |            |        |--ppi        :issip,issipp,issippi
         |           |--ppi                  : ip,ipp,ippi
         |----> s--> |--si--> |--ssippi  s  : ssissippi
         |           |        |--ppi        :ssippi,ssip,ssipp
         |           |--i -- > |--ssippi   si : sissippi
         |                     |--ppi          sip,sip,sippi
         |---->     |--pi                     p :  ppi,p,pp
                     |--i                          p,pi
```

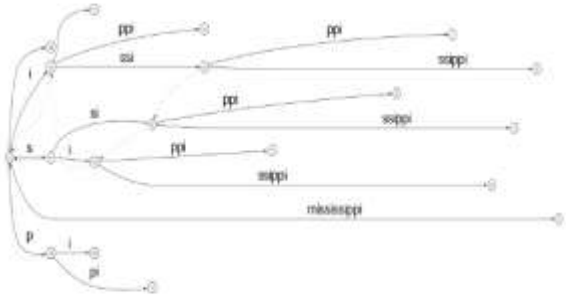


Figure 2: Suffix tree structure for 'mississippi'

The suffix tree clustering algorithm takes a document as a string. The suffix tree can be easily identified the shared common phrases and use information for making the clusters. The suffix tree data structure is the heart of the suffix tree clustering suffix tree is constructed from set of strings. The sentences from the documents inserted in to the suffix tree as a word, not as a character.

Definition A Suffix Tree ST for an m character string S is rooted directed tree with exactly m leaves numbered 1 to m. Each internal node, other than the root has at least two children. And each edge is labeled with non-empty substring of S. No two edges out of a node can have edge labels beginning with the same character.

Algorithm steps performing the Suffix Tree Clustering[8]:

Step 1: Collection of documents
Step 2: Preprocessing (Document Cleaning)
Step 3: Identify the base clusters
Step 4: Merges the base clusters
Step 5: Labeling

Step 1: Collection of documents

The collection of documents is the very first to perform the searching. The documents are collected in the dataset.

The collected documents can be either text documents or the web documents, but the tool performs the clustering on the text documents. Thereafter the document cleaning should be done.

Step 2: Preprocessing (Document Cleaning):

Preprocessing is the step that performs the document cleaning. . In Document cleaning, data is cleaned from the missing values, smoothing noisy data and inconsistencies.

Data cleaning is the preprocessing of the data, through which data is cleaned and processed that is input to the next step to the Suffix Tree structure. Preprocessing includes the steps such as:

- Tokenization
- Stop-word removal
- Stemming algorithm

Tokenization: Tokenization is the preprocessing step, in which sentences are divided into tokens. . Tokenization is the process of identify the word and sentences boundaries in the text. The simplest form of tokenization is the white space character as a word delimiters and selected punctuation mark such as '.','?'and '! '. Each word assigns a token id.

Stop-word removal: There are many words in the document that contain no information about the topic. Such words don't have any meaning or no use while creating the suffix tree structure. Stop words are also referred to the as function words that have their own identifiable meaning. Such words that occur in the stop list are: and, but, will, have etc. The list of stop word is store in the database.

Stemming Algorithm: In the stemming procedure all words in the text document are replaced with their respective stem. A stem is a portion of a word that would be left after removing the affixes (suffixes and prefixes). Different form of words can be reduced into one base form by using the stemmer. Lots of stemmer created for the English language. The process of stemmer development is easy. There is lot of stemmers available for English language such as: Porter stemmer, Paice stemmer and Lovins Stemmer. For example: connected, connecting, interconnection is transformed into word connect.

After applying the preprocessing step the documents will be cleaned and ready for the identifying the base clusters, which is the next step for the suffix tree clustering algorithm.

Step 3: Identify the base clusters:

The suffix tree clustering[12], [13] work in two main phases first, is the identification of base clusters and second, is the merging the base clusters. In base cluster identification phase of the suffix tree clustering algorithm, the base clusters are identified. The base clusters consist of the words and phrases contain in the documents. The suffix tree constructed in linear in time and size. The suffix tree has advantage that it can find the phrases of any length and it is fast and efficient in finding the phrases that is shared by two or more documents.
For example, there are three documents through which we can define the base clusters and merging of the clusters.
The documents are:

Doc1: a cat ate cheeseing.
Doc2: mouse and cat ate cheese.
Doc3: cat ate mouse too

The base clusters after applying the document cleaning process:

Base clusters identification of the above documents are:

Words clusters	documents
cat	1, 2, 3
ate	1, 2, 3
cheese	1, 2
mouse	2, 3

phrase clusters	documents
cat ate	1, 2, 3
ate cheese	1, 2

A suffix tree of string S containing all the suffixes of S. the documents are treated as string of words. The suffixes in the suffix tree containing one or more words. Terms to be used for suffix tree:

- A suffix tree is a rooted tree.
- Each internal node has at least two children.
- Each edge is labeled with non-empty substring of S. the label of a node is defined to be the concatenation of edge label on the path from root to that node.
- No two edges out of same node can have edge labels that begin with the same word.

For each suffix s of S there exist suffix nodes whose label equal to s.

Each base cluster assigned a score which is the function that number of documents it contains and the words that makes up its phrases. The score function calculated for the base clusters, balance the length of phrases, coverage of all candidate clusters (the percent of all collection of document it contain) and the frequency of phrase term in the total collection of documents. A candidate node becomes a base cluster if and only if it exceeds a minimal base cluster score.

The score function is defined by the formula as:

$$s(m) = |m| \cdot f(|m_p|) \cdot \sum tfidf(w_i) \quad (6)$$

Where:

S (m) - the score of candidate m

|m| - number of phrase terms

f (|m_p|) - phrase length adjustment

tfidf(w_i) - term frequency adjustment

tfidf is Term Frequency and Inverse Term Frequency measures for assigning weight to terms. The formula which is used to calculate the tfidf:

$$tfidf(w_i, d) = (1 + \log(tf(w_i, d))) \cdot \log(1 + N / df(w_i)) \quad (7)$$

Where:

tf (w_i, d) - number of terms w_i occurred in document d.

N - total number of documents

df (w_i) - number of documents term w_i appear in

Step 4: Merging (combining) base clusters:

Phrases are shared between one or more documents. The next step of the suffix tree clustering is the merging of the base cluster. The base clusters are merged (words and phrases). After merging the clusters the similarity is calculated of the base clusters.

The similarity can be calculated on the basis of the similarity measures. There are two similarity measures one is the binary similarity and other is the cosine similarity. In the tool use both of the similarity measures to calculate the similarity between the documents.

Cosine similarity measures:

Cosine similarity measure is used to calculate the similarity between two documents. There is several ways to compute the similarity between documents. We use the binary similarity and cosine similarity to compute the similarity between the documents. The similarity between the documents is known as the small distance in one cluster. Documents are represented by the vectors where each attribute represent the frequency of word with a particular word occur in the document. The equation which used to calculate the similarity

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \quad (8)$$

Cosine of two vectors can be calculated by using the Euclidean dot product:

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \cos\theta \quad (9)$$

A and B are the two vectors of attributes. For text matching the attribute vector of A and B are term frequency vectors of the documents.in case of information retrieval the cosine similarity ranges from 0 to 1 and the term frequency cannot be negative. Each word in the texts defines a dimension in Euclidean space and the frequency of each word corresponds to the values in the dimension.

Binary similarity measures:

In binary similarity measures we use the formula for clustering the text documents. the binary similarity is used between base clusters on the overlap of their document sets. For example given two base clusters Bm and Bn with the size |Bm| and |Bn| and |Bm ∩Bn| representing the number of documents common to both base clusters. the similarity of Bm and Bn to be 1 if

$|Bm \cap Bn| / |Bm| > 0.5$ and, (9)

$|Bm \cap Bn| / |Bn| > 0.5$ (10)

Otherwise the similarity is defined to be 0.

Step 5: Labeling:

Labeling is used for label the clusters by the words and phrases of the documents and the suffixes identified while creating the base clusters. The suffix tree structure is labeled by the suffixes of the documents that are identified during the document cleaning and other process of creation of the suffix tree.

Auto assemblage is the tool which is fellow the all the steps of the suffix tree clustering algorithm. All the steps define by the screen shots in the next section.

V. Steps of Suffix tree Clustering algorithm (Diagram)

The diagram defines the steps of suffix tree clustering algorithm such as:

- Datasets collection as input
- Document cleaning
- Identify the base clusters
- Merging base clusters

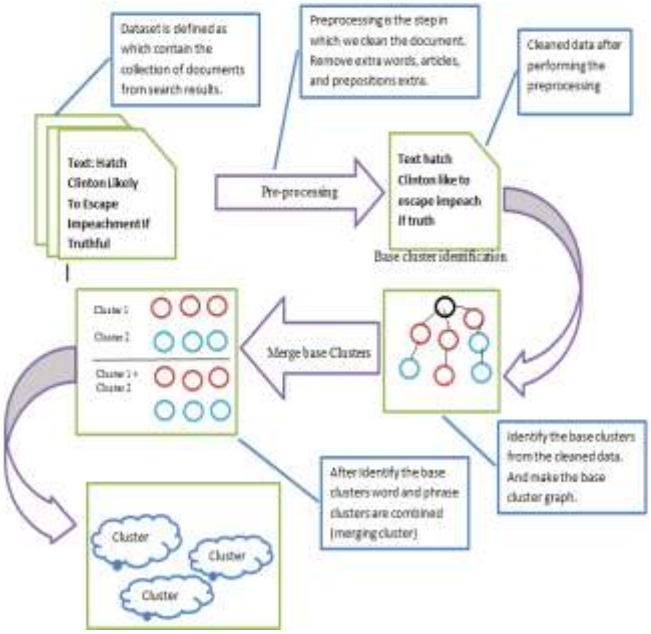

Dataset is defined as which contain the collection of documents from search results.
Preprocessing is the step in which we clean the document. Remove extra words, articles, and prepositions extra.
Cleaned data after performing the preprocessing
Text: Hatch Clinton Likely To Escape Impeachment If Truthful
Pre-processing
Text hatch Clinton like to escape impeach If truth
Base cluster identification
Cluster 1
Cluster 2
Cluster 1 + Cluster 2
Merge base Clusters
Identify the base clusters from the cleaned data. And make the base cluster graph.
After identify the base clusters word and phrase clusters are combined (merging cluster)
Cluster
Cluster
Cluster

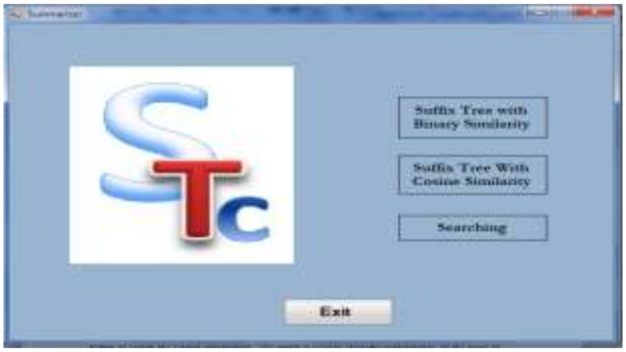
Figure 3: steps for suffix tree clustering

VI. Tool Description (Auto assemblage version 1.0.0)

The tool auto assemblage is implemented in the software Microsoft visual studio 2008 as a front end and datasets are stored in the database Microsoft SQL server 2000 as a back end. All the steps are defined by the screen shots that are implemented during the thesis work.

Screen shots:

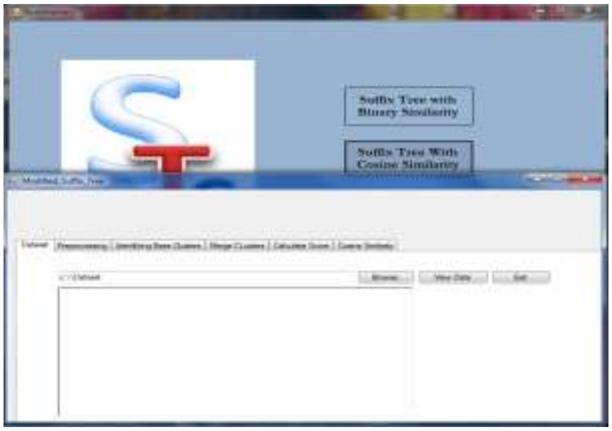
Step 1:


Summarizer
STc
Suffix Tree with Binary Similarity
Suffix Tree With Cosine Similarity
Searching
Exit

Design view1: summeraizer

The summarizer screen is summarizing the contents it contains the Description about the Suffix Tree Clustering with binary similarity and Suffix Tree Clustering with cosine similarity. And the searching button to search the related information. The graph is used to show the performance on the basis of the similarity measures between the clusters.

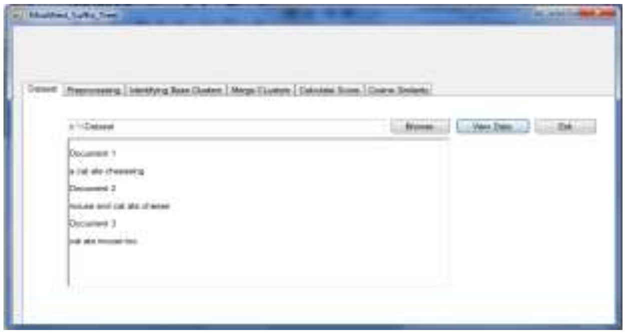
Step 2:



Design view 2: Suffix Tree with Cosine Similarity

After clicking on the Suffix Tree with Cosine Similarity other screen which is opened is the suffix tree is opened that consist of the steps of the suffix tree clustering algorithm. The algorithm calculates the similarity using of the binary similarity. The window has steps: Dataset, Preprocessing, Identify the base clusters, merging clusters, calculate score with term frequency and inverse document frequency and similarity.

Step 3:



c:\Dataset

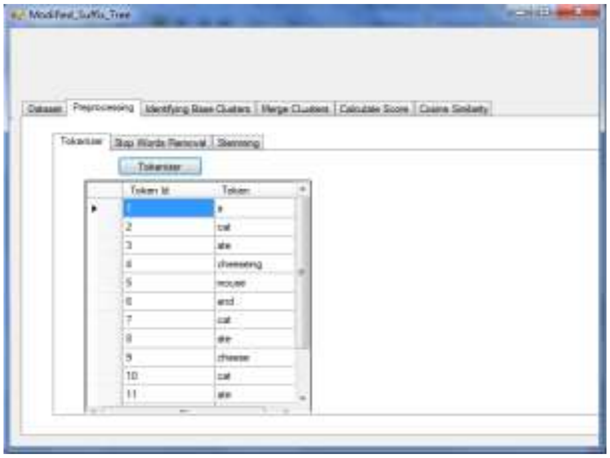
Document 1
a cat ate cheeseing
Document 2
mouse and cat ate cheese
Document 3
cat ate mouse too

Design view 3: Dataset

This step defines the datasets that are stored in the database. In this step the datasets are display only.

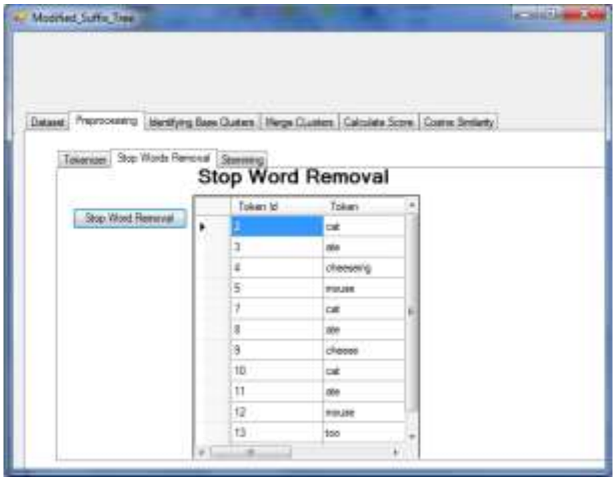
Step 4:

This step defines the all the necessary steps for the document cleaning (preprocessing). Such as: Tokenization Stop-words removal and stemming algorithm.



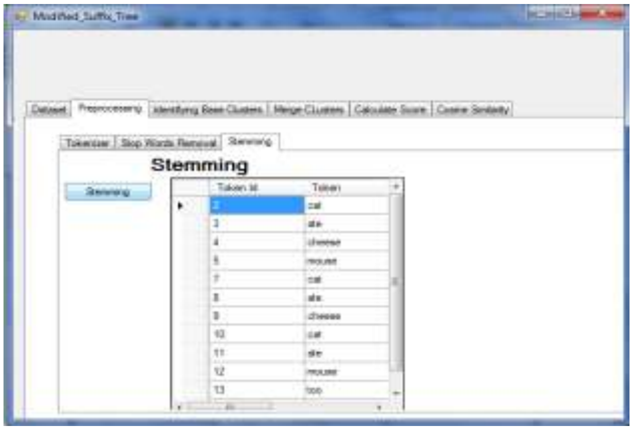
Token Id	Token
1	a
2	cat
3	ate
4	cheeseing
5	mouse
6	and
7	cat
8	ate
9	cheese
10	cat
11	ate

Design view 4: Tokenization



Token Id	Token
2	cat
3	ate
4	cheeseing
5	mouse
7	cat
8	ate
9	cheese
10	cat
11	ate
12	mouse
13	too

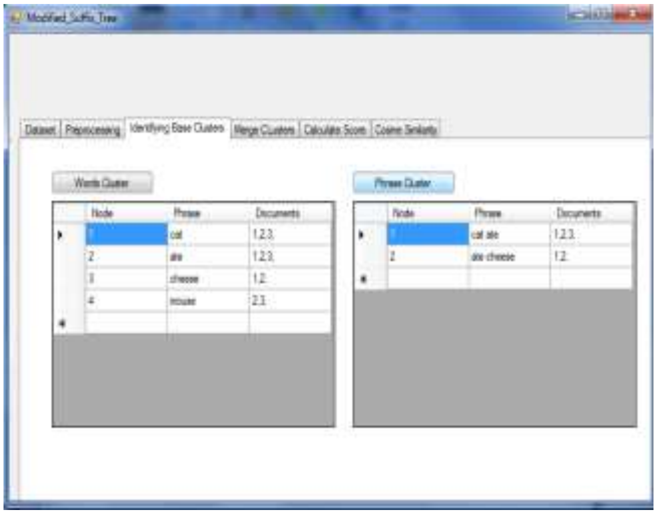
Design view 5:Stop-word removal



Token Id	Token
2	cat
3	ate
4	cheese
5	mouse
7	cat
8	ate
9	cheese
10	cat
11	ate
12	mouse
13	too

Design view 6: Stemming algorithm

Step 5:



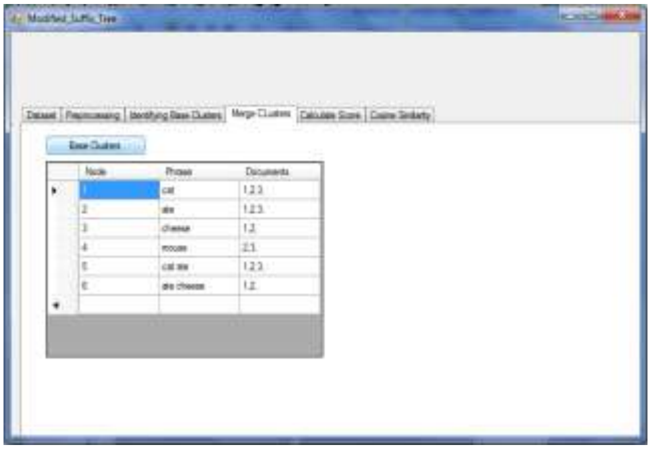
Node	Phrase	Documents
1	cat	1,2,3,
2	ate	1,2,3,
3	cheese	1,2,
4	mouse	2,3,

Node	Phrase	Documents
1	cat ate	1,2,3,
2	ate cheese	1,2,

Design view 7: Base clusters

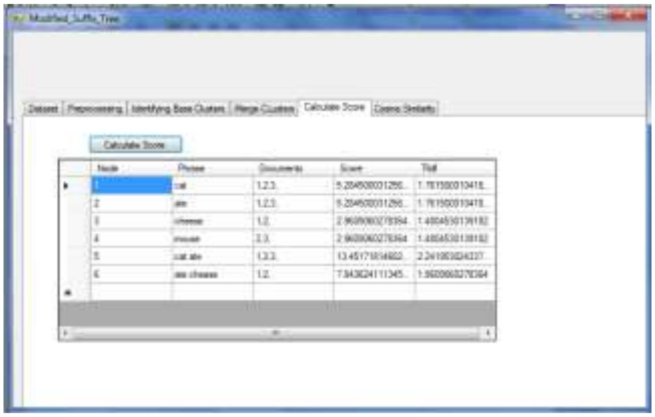
There are two main phases of the suffix tree clustering that is identify the base clusters and merging the base clusters. In the base clusters, we have to find out the common words and phrases in the documents. The above screen shows the word clusters and phrase clusters.

Step 6:



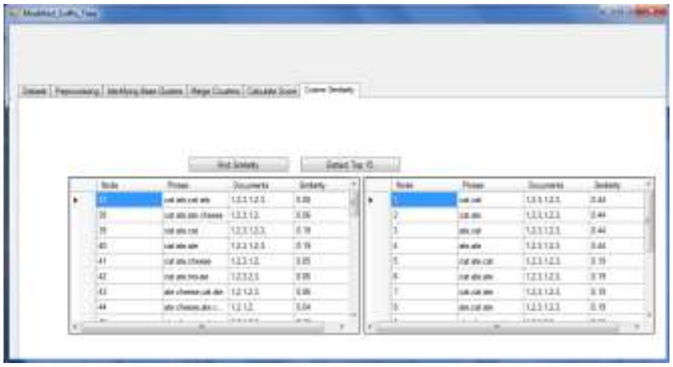
Node	Phrase	Documents
1	cat	1,2,3,
2	ate	1,2,3,
3	cheese	1,2,
4	mouse	2,3,
5	cat ate	1,2,3,
6	ate cheese	1,2,

Design view 8 : Merging the clustering



Node	Phrase	Documents	Score	TfIdf
1	cat	1,2,3,	5.2045000031256...	1.7615000010418...
2	ate	1,2,3,	5.2045000031256...	1.7615000010418...
3	cheese	1,2,	2.9609060278384...	1.4804530139182
4	mouse	2,3,	2.9609060278384	1.4804530139182
5	cat ate	1,3,3,	13.4517181480...	2.2410030243237...
6	ate cheese	1,2,	7.9436241113456...	1.9609060278384

Design view 9 : Score calculation

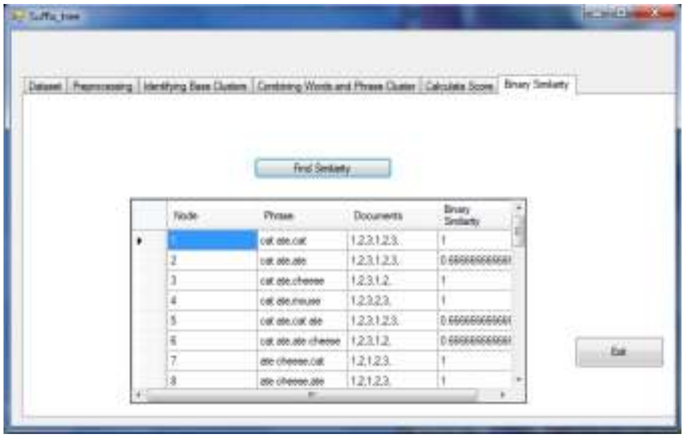


Node	Phrase	Documents	Similarity
37	cat ate cat ate	1,2,3,1,2,3,	0.02
38	cat ate ate cheese	1,2,3,1,2,	0.06
39	cat ate cat	1,2,3,1,2,3,	0.19
40	cat ate ate	1,2,3,1,2,3,	0.19
41	cat ate cheese	1,2,3,1,2,	0.05
42	cat ate mouse	1,2,3,2,3,	0.08
43	ate cheese cat ate	1,2,1,2,3,	0.06
44	ate cheese ate c...	1,2,1,2,	0.04

Node	Phrase	Documents	Similarity
1	cat cat	1,2,3,1,2,3,	0.44
2	cat ate	1,2,3,1,2,3,	0.44
3	ate cat	1,2,3,1,2,3,	0.44
4	ate ate	1,2,3,1,2,3,	0.44
5	cat ate cat	1,2,3,1,2,3,	0.19
6	cat ate ate	1,2,3,1,2,3,	0.19
7	cat cat ate	1,2,3,1,2,3,	0.19
8	ate cat ate	1,2,3,1,2,3,	0.19

Design view10 : Cosine similarity measures

Step 7:

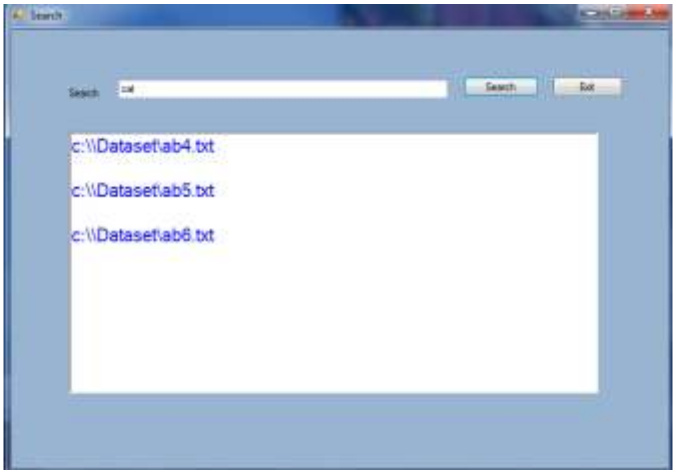


Node	Phrase	Documents	Binary Similarity
1	cat ate.cat	1,2,3,1,2,3,	1
2	cat ate.ate	1,2,3,1,2,3,	0.666666666666
3	cat ate.cheese	1,2,3,1,2,	1
4	cat ate.mouse	1,2,3,2,3,	1
5	cat ate.cat ate	1,2,3,1,2,3,	0.666666666666
6	cat ate.ate cheese	1,2,3,1,2,	0.666666666666
7	ate cheese.cat	1,2,1,2,3,	1
8	ate cheese.ate	1,2,1,2,3,	1

Design view 11 : Binary similarity measures

After performing all the steps the last step is the searching that uses the clusters that is created during the suffix tree clustering.

Step 8:



Search cat

c:\\Dataset\ab4.txt

c:\\Dataset\ab5.txt

c:\\Dataset\ab6.txt

Design view 12 : Search the data

The datasets that are used is display in the step2. In this step the user enter the string that is to be searched.

VII. Conclusion

The paper first defines the brief introduction about the document clustering and different document clustering techniques such as partitioned and hierarchical document clustering. The next step is the suffix tree clustering algorithm which is the base of the paper and then defines their algorithmic steps that perform the clustering process. After that the diagram is displayed that defines the algorithm steps. At last the tool which is implemented is described with the screen shots.

References

[1] Kale, U. Bharambe, M. Sashi Kumar, "A New Suffix Tree Similarity Measure and Labeling for Web Search Results Clustering", *Proc.* Second International Conference on Emerging Trends in Engineering and Technology, ICETET-09, p.856-861.

[2] (2012).L. B. Ayre, "Data mining for information Professional".

[3] V. M. A. Bai and Dr. D. Manimegalai, "An Analysis of Document Clustering Algorithm", in ICCCCT-10, IEEE 2010, p.402-406.

[4] S.Na,G. yongand L. Xumin, "Research on K-means Clustering Algorithm",Third internation Symposium on intelligent Information Technology and security informatics,2010 IEEE,p. 63-67.

[5] (2012). "K-Means Clustering Tutorials" http:\\people. revoledu.com\kardi\ tutorial\kMean\.

[6] G. Zhang, Y.Liu, S.Tan, and X.Cheng, "A Novel Method for Hierarchical Clustering of Search Result", 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Workshops.

[7] H.Sun, Z.Liu and L.Kong, "A Document Clustering Method Based on Hierarchical Algorithm with Model Clustering", 22nd International Conference on Advanced Information Networking and Application-Workshops. IEEE 2008, p.1229-1233.

[8] O. Zamir and O. Etzioni, "Web Document Clustering: A Feasibility Demonstration", in *Proc.* the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, 1998, p. 46-54.

[9] H. Chim and X. Deng, "Efficient Phrase-Based Document Similarity for Clustering," IEEE Transaction on Knowledge and Data Engineering, vol. 20, no. September 2008, pp. 1217-1229.

[10] (2011) home page on CS.[Online].Avalable: http://www.cs.gmu.edu/cne/modul e/dau/stat/ clustgalgs/clust5_bdy.html.

[11] (2011) Available: http://www.allisons.org.

[12] S.osiuski and D.Weiss, "A Concept-Driven Algorithm for Clustering Search Results", IEEE 2005.

[13] Rafi, M.Maujood, M.M.Fazal, S.M.Ali, "A Comparison of Two Suffix Tree Based Document Clustering Algorithm", in *Proc*. IEEE 2010NU-FAST, Karachi, Pakistan.

[14] J.Han and M.Kamber, "Data Mining Concepts and Techniques", 2nd Edition, 2006 Elsevier.

Pushplata received the Bachelor degree in Computer Science and Engineering from Maharishi Dayanand University Rohtak, India in 2010. She is doing her Master's in Computer Engineering from Maharishi Dayanand University Rohtak (Manav Rachna College of Engineering). Her Research interest is Data Mining (Clustering) including theory and techniques of the data mining.

Mr. Ram Chatterjee received his Master's in Master of Computer Application and M.Tech (Computer Science and Engineering) from CDAC, Noida. He is working as Assistant Professor in Manav Rachna College of Engineering, Computer Science Department, and Faridabad - 121004. Haryana, INDIA. His interest area is Data mining and Software Engineering.