Saudi Journal of Engineering and Technology (SJEAT)
Scholars Middle East Publishers
Dubai, United Arab Emirates
Website: http://scholarsmepub.com/

ISSN 2415-6272 (Print)
ISSN 2415-6264 (Online)

Ontology Based Automatic Text Mining Using TF and IDF Algorithms for Summarization of Multiple Files

Chinmayee C[1*], Dr. S Meenakshi Sundaram[2], Keerthana N S[1], Manikya S[1], Nitya Hegde M[1]
[1]Department of CSE, GSSS Institute of Engineering and Technology for Women, Mysuru, Karnataka, India
[2]Professor and Head, Department of CSE, GSSS Institute of Engineering and Technology for Women, Mysuru, Karnataka, India

Original Research Article

***Corresponding author**
Chinmayee C

Article History
Received: 12.06.2018
Accepted: 22.06.2018
Published: 30.06.2018

DOI:
10.21276/sjeat.2018.3.6.8

Abstract: In the present world, due to tremendous development in technology, a huge amount of information is available everywhere. Therefore, it is difficult for the users to understand the main content of the entire document as it takes a lot of time. In this work we use extractive text summarization which uses a method to give the version of summary for one or more file or document. Here we give an approach that maps sentences to nodes of a hierarchical ontology. Ontology explains what exists in a particular domain. For the ontology creation, vocabularies are collected. It is used as background knowledge and helps to find the related meaning of the terms which occur in the source documents. Text mining is the technique from which high quality information is derived from text. Clustering is a significant task. The clustering method groups similar or related terms into a single group. In the first stage, data collection takes place. The pre-processing stage includes stemming and stop words removal.TF-IDF process occurs after which clustering takes place. In the ontology creation, first the determination of the main sub topics of the article of interest is done. We classify sentences to nodes which have a predefined hierarchical ontology. Each ontology node has bag-of-words from a web search. We represent sentences by sub trees that permit to apply measures of similarity and find relations between sentences. The ontology used in this work is not domain-specific; it does not require labelled data. this work can be extended to topics focused on summarization framework to news articles or blogs and to also to various machine learning approaches

Keywords: Ontology, Text Summarization, TF-IDF, Files, Documents, Extract, Summary.

INTRODUCTION

In text summarization one document is created from one or more textual source document. It will be smaller in size but most of the original information is retained. The main goal is to create summaries which are similar to abstracts created by humans. Extracts contain units of texts which are selected from the source documents. It evaluates based on the key information and gives summary features like term prominence, rhetorical structure, graph-theoretic and semantic features are used. The authors use the ontologies for expansion of query, for representing sentences through bag-of-words.

The bag-of-words contain words that are equivalent to ontology concepts. In term –based mapping, the concept of generalization options offered by WordNet relations are exploited to find the concepts that are most informative.

In Ontology –based Text Summarization for business news articles, the authors construct anontology manually for small domain of news articles. They use category labels to score paragraphs. That score is increased together with its parent categories if paragraph contains that label. Category with highest scores are selected as main topics. Paragraphs are selected until desired length. This approach ranks paragraphs and not sentences. It considers only category labels and synonyms are also not recognized.

RELATED WORK

“An Ontology-Based Text Mining method to construct D-Matrix for Fault Detection and Diagnosis using graph comparison Algorithm” was proposed by Ms. Madhuri M Varma *et al.,* [1] and published in International Journal of Innovative Research in Information Security in May 2015 . Here D-Matrix is used to catch the different system level fault diagnostic information consisting of conditions between recognizable symptoms and failure modes connected with a system. This system comprises the developments of D-Matrix from the repair verbatim data. After the creation of the D-Matrices from the different datasets, the graph was generated for each D-Matrix. A comprehensive D-Matrix as developed using text mining method which was based on ontology by which they can store information obtained during fault recognizing and fault solving practices.

"Effective Pattern Discovery for Text Mining" was proposed by Ning Zhong *et al.,* [2] and published in IEEE Transactions on Knowledge and Data Engineering in Jan. 2012. This paper has presented an innovative and effective pattern discovery technique which includes the processes of pattern deploying and pattern evolving to improve the effectiveness of using and updating discovered patterns for finding relevant and interesting information. They focused on the development of a knowledge discovery model to effectively use and update the discovered patterns and apply it to the field of text mining. Their proposed technique uses two processes, pattern deploying and pattern evolving to refine the discovered patterns in text documents. Their experimental results show that the proposed model outperforms not only other pure data mining-based methods and the concept-based model, but also term-based state-of-the-art models such as BM25 and SVM-based methods.

"Discovering Low-Rank shred concept space for Adapting Text Mining Models" was proposed by Bo Chen *et al.,* [3] and published in IEEE Transactions on Pattern Analysis and Machine Intelligence in June 2013. In this method, they conducted the domain adaptation task both in the original feature space as well in the transformed Reproducing Kernel Hilbert Space (RKHS) using kernel tricks. They proposed a framework for adapting text mining models that discovers low-rank shared concept space. Their major characteristics of this concept space was that it explicitly minimizes the distribution gap between the source domain with sufficient labelled data and the target domain with only unlabelled data. In their method, they parameterize the shared space by a linear transformation and find the optimal solution by considering the combination of two criteria: The empirical loss on the source domain and the embedded distribution gap between the source domain and the target domain.

"A Fuzzy Self-Constructing Feature clustering Algorithm for Text Classification" was proposed by Jung-Yi Jiang *et al.,* [4] and published in IEEE Transactions on Knowledge and Data Engineering in March 2011. Here each cluster is characterized by a membership function with statistical mean and deviation. Feature-Clustering was a powerful method to reduce the dimensionality of feature vectors for text classification. This algorithm was an incremental clustering approach to reduce the dimensionality of the features in text classification.

"An Ontology-Based Text Mining Method to cluster proposals for Research Project Selection" was proposed by Jian Ma *et al.,* [5] and published in IEEE Transactions on Systems, Man and Cybernetics in May 2012. This paper has presented a novel Ontology-Based Text-Mining approach to cluster proposals based on their similarities in research areas. This method is efficient with both English and Chinese texts. The proposed method can be used to expedite and improve the proposal grouping process in the NSFC and elsewhere. It uses the data collected from a research social network and extends the functions of the Internet-Based Science Information System.

"An efficient concept Based Mining Model for Enhancing Text Clustering" was proposed by Shady Shehata *et al.*, [6] and published in IEEE Transactions on knowledge and Data Engineering in October 2010. Here a new concept-based mining model that analyses terms on the sentence, document and corpus level is introduced. The concept-based mining model can effectively discriminate between unimportant terms with respect to sentence semantics and terms which hold the concepts that represent the sentence meaning. A new concept-based mining model composed of four components, was proposed to improve the text clustering quality. By exploiting the semantic structure of the sentences in documents, a better text clustering result is achieved.

"Computing Multi-Dimensional Trust by Mining E-Commerce Feedback comments" was proposed by Xiuzhen Zhang *et al.,* [7] and published in IEEE Transactions on Knowledge and Data Engineering in Jan. 2007. Here an algorithm for mining feedback comments dimension ratings and weights, combining techniques of natural language processing, opinion and topic mining. Reputation-Based trust models are widely used in E-Commerce applications and feedback ratings are aggregated to compute sellers reputation trust scores. Based on the observation that buyers often express opinions openly in free text feedback comments. They proposed CommTrust for trust evaluation by mining feedback comments. They proposed to compute comprehensive multi-dimensional trust profiles for sellers by uncovering dimension ratings embedded in feedback comments. Extensive experiments on feedback comments for eBay and Amazon sellers demonstrate that their approach computes trust scores highly effective to distinguish and rank sellers.

PROPOSED WORK

It uses extraction summarization approaches to perform the automatic summarization. Extractive methods work by selecting words, phrases or sentences in the original text to form the summary.

Pre processing

In the first step splitting of the sentences into words takes place following white space as the separator. The next step is stemming. Stemming is the methodology to get the root of the particular word in the document. The process is continued by removing stop word. Stop word removal is by comparing each word in the sentence. Examples are articles, conjunctions and prepositions.

TF-IDF

TF-IDF stands for term frequency-inverse document frequency and the tf-idf weight is a weight used in information retrieval and the text mining. It tells how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document. TF: Term Frequency tells how frequently a term occurs in a document. Term frequency is often divided by the document length (that is total number of terms in the document) TF(t)=(number of times term t appears in a document)/(total number of terms in the document) IDF: Inverse Document Frequency, which measures how important a term is while computing TF, all terms are considered equally important. But certain terms, such as "is", "of" and "that" may appear lot of times but they have little importance. IDF(t)=log-e(total number of documents/ number of documents with term t in it) Tf-idf weight=tf*idf.

Clustering

Clustering is the process of grouping a similar or related terms into a single group. Identification of concepts from text takes place. These are called key concepts of the target domain. Steps for clustering are: a) A vector of high dimensional concept is given. b) Concepts for clustering are generated (that is terms and related terms). c) Based on the concepts, the initial clusters are constructed (that is terms and related terms). d) The cluster is disjointed to identify the best initial cluster and by the goodness score calculation, the document is kept only in that cluster. e) The cluster is built.

Ontology Creation

For these vocabularies are collected .Next, those words are put by the data model of ontology. In the first step, determination of the main subtopics of the article of interest is done. This is obtained by the comparison of words of articles with terms in the ontology. The non-existing words in the ontology are ignored. The number of times the word appears in the ontology is recorded. In the tree structure, each node includes the nodes children. If the count of the node increases, the ancestors count will also increased. From this type of design, the root of the ontology always gets the highest score and the second level nodes represented by subtopics gets different score. The second-level nodes with higher counts are selected as the main subtopics of the article.

Refined Graph

The graph is filtered by the calculation of frequency threshold value. The depth limit of the ontology graph is decided by this threshold value. Multiple depth limit may be contained in the graph. For refining the graph, user can select the threshold value.

Summarized Text

Here the condensed version that is summary is given collecting the results of the refined graph.

SYSTEM ARCHITECTURE

Fig-1 above shows the system architecture that includes corpus selection which is uploading multiple text files as input. Pre processing takes the input and splits sentences into words. Feature extraction involves Stemming and Stop words removal. Frequency calculation includes TF and IDF. Deviation Estimation includes Similarity and Clustering. In Ontology construction, vocabularies are collected. Finally summary is given for the initial document.

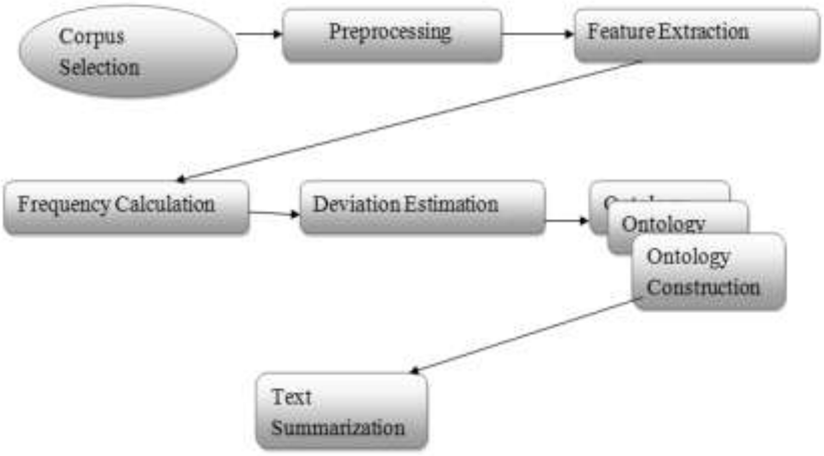


Fig-1: System Architecture

Advantages of the proposed system

Paragraphs are selected until the desired summary length is reached. Feature selection accuracy is improved.

METHODOLOGY ADOPTED

The algorithm used in this work is given below:

A consonant in a word is a letter other than A, E, I, O or U and other than Y preceded by a consonant. A consonant will be denoted by c, a vowel by v. A list ccc.. of length greater than 0 will be denoted by C, and a list vvv… of length greater than 0 will be denoted by V. Any word, or part of word, therefore has one of the four forms:

CVCV ... C
CVCV ... V
VCVC ... C
VCVC ... V

These may all be represented by the single form

[C]VCVC…[V]

Where, the square brackets denote arbitrary presence of their contents. Using (VC)^m to denote VC repeated m times, this may again be written as

[C](VC)^m[V].

M will be called measure of any word or word part when represented in this form. The case m=0 covers the null word. Here are some examples:

m=0	TR, EE, TREE, Y, BY.
m=1	TROUBLE, OATS, TREES, IVY.
m=2	TROUBLES, PRIVATE, OATEN, ORRERY.

The rules for removing a suffix will be given in the form (condition)S1->S2

This means that if a word ends with the suffix S1, and the stem before S1 satisfies the given condition, s1 is replaced by S2.The condition is usually given in terms of m, e.g.

(m>1)EMENT->

Here S1 is ‘EMENT’ and S2 is null. This would map REPLACEMENT to REPLAC, since REPLAC is a word part for which m=2.

The ‘condition’ part may also contain the following:

*S	-	the stem ends with S (and similarly for the other letters).
v	-	the stem contains a vowel.
*d	-	the stem ends with a double consonant (e.g. -TT, -SS).
*o	-	the stem ends cvc, where the second c is not W, X or Y (e.g. -WIL, -HOP).

<u>Step 1a</u>

• SSES -> SS	caresses -> caress
• IES -> I	ponies -> ponities -> ti
• SS -> SS	caress -> caress
• S ->	cats -> cat

<u>Step 1b</u>

• (m>0) EED -> EE	agreed -> agree
• (*v*) ED ->	plastered -> plaster
• (*v*) ING ->	motoring -> motor

If the second or third of the rules in step 1b is successful, the following is done

AT -> ATE	conflat(ed) -> conflate
BL -> BLE	troubl(ed) -> trouble
IZ -> IZE	siz(ed) -> size

- Step 1 deals with plurals and past participles.

<u>Step 2:</u>

• (m>0) ATIONAL -> ATE	Relational->Relate
• (m>0) IZER -> IZE	Digitizer->Digitize
• (m>0) ALLI -> AL	Radically->Radical

<u>Step 3</u>:

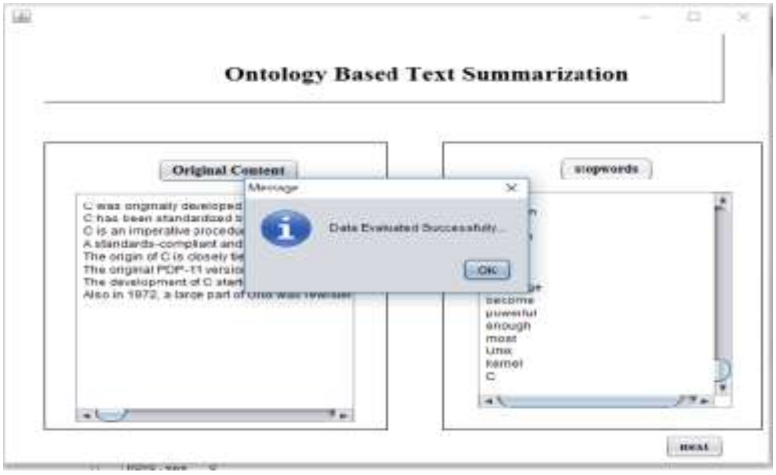
• (m>1) ICATE-> IC	Triplicate-> Triplic
• (m>1) EMENT->	Replacement -> Replac

<u>Step 4:</u>

(m>1) E ->	probate -> probat

RESULTS AND DISCUSSION

Figures-2 to 14 given below describe various results obtained in this work. Fig. 2 shows instances of uploading multiple files and performing Stop words process. In this process it removes articles, conjunctions and prepositions. A sample case for successful data evaluation is shown in the figure.



Ontology Based Text Summarization

Original Content

C was originally developed
C has been standardized b
C is an imperative procedu
A standards-compliant and
The origin of C is closely tie
The original PDP-11 versio
The development of C start
Also in 1972, a large part of Unix was rewritten

Message

Data Evaluated Successfully...

OK

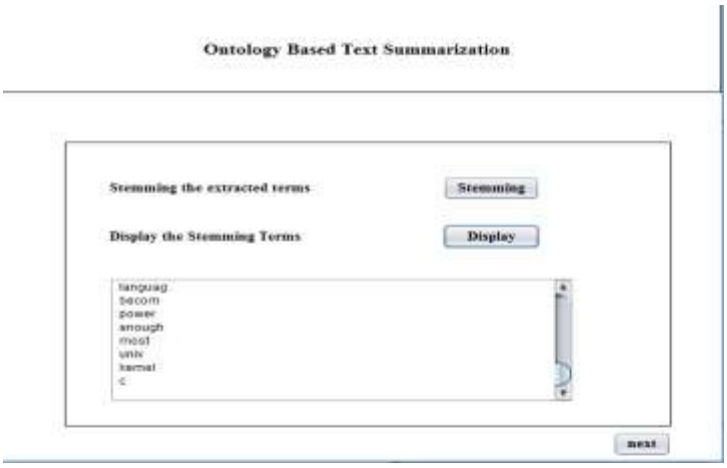
stopwords

become
powerful
enough
most
Unix
kernel
C

next

Fig-2: Uploading Multiple Files and performing Stop words Process.

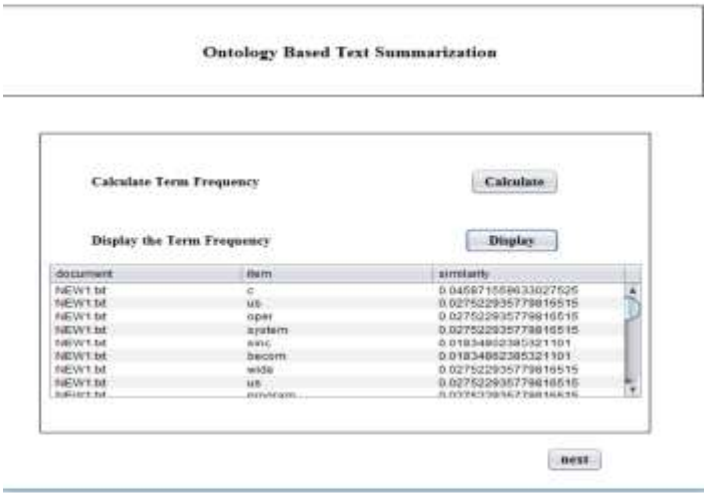
Fig-3 shows the process of stemming. Here suffixes are removed and the root words are given. The upper case letters are converted into lower case letters.


Ontology Based Text Summarization
Stemming the extracted terms
Stemming
Display the Stemming Terms
Display
next

```
languag
becom
power
enough
most
unix
kernel
c
```

Fig-3: Process of Stemming on Uploaded files.

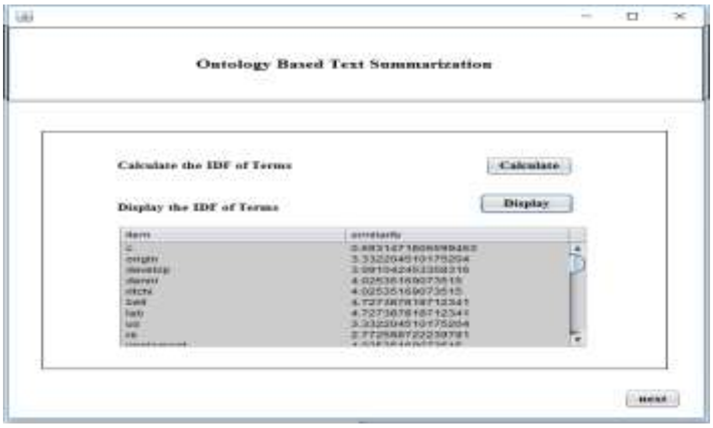
Fig-4 shows the Term Frequency Calculation. It calculates how frequently a term appears in the document.



document	item	similarity
NEW1.txt	c	0.045871559633027525
NEW1.txt	us	0.027522935779816515
NEW1.txt	oper	0.027522935779816515
NEW1.txt	system	0.027522935779816515
NEW1.txt	sinc	0.01834862385321101
NEW1.txt	becom	0.01834862385321101
NEW1.txt	wide	0.027522935779816515
NEW1.txt	us	0.027522935779816515
NEW1.txt	program	0.027522935779816515

Fig-4: Calculating Term Frequency

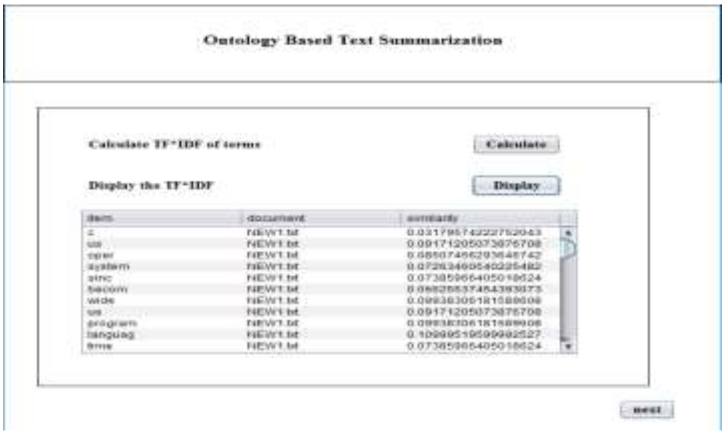
Fig-5 shows the calculation of Inverse Document Frequency. It measures how important a term is while calculating TF.



item	similarity
c	3.4831471806599453
origin	3.332204510175204
develop	3.091042453358316
dennis	4.02535169073515
ritchi	4.02535169073515
bell	4.727387818712341
lab	4.727387818712341
us	3.332204510175204
os	2.772588722239781
[illegible]	[illegible]

Fig-5: Calculating Inverse Document Frequency.

Fig-6 shows the calculation of TF*IDF weight. This weight is used in information retrieval from initial document .



Ontology Based Text Summarization

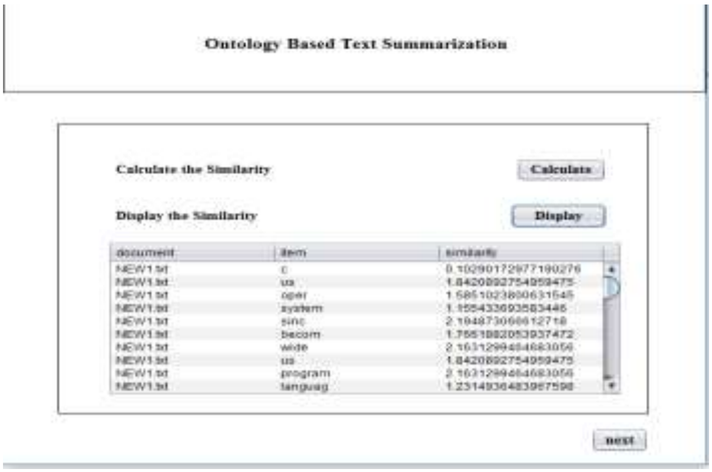
Calculate TF*IDF of terms

Display the TF*IDF

item	document	similarity
c	NEW1.txt	0.03179574222752043
us	NEW1.txt	0.09171205073876708
oper	NEW1.txt	0.08507466293648742
system	NEW1.txt	0.07263469540225482
sinc	NEW1.txt	0.07385966405018624
becom	NEW1.txt	0.08825537484383073
wide	NEW1.txt	0.09938306181588608
us	NEW1.txt	0.09171205073876708
program	NEW1.txt	0.09938306181588608
languag	NEW1.txt	0.10999519599992527
time	NEW1.txt	0.07385966405018624

Fig-6: Calculation of both TF and IDF

Fig-7 shows the calculation of similarity. It finds the similarity between different words.



Ontology Based Text Summarization

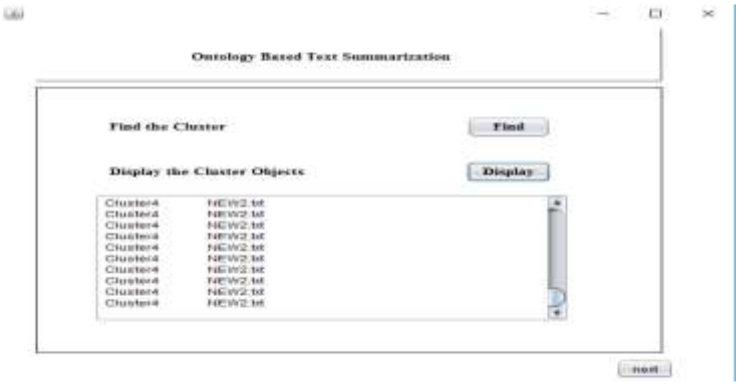
Calculate the Similarity

Display the Similarity

document	item	similarity
NEW1.txt	c	0.10290172977190276
NEW1.txt	us	1.8420892754959475
NEW1.txt	oper	1.5851023800631545
NEW1.txt	system	1.155433693583446
NEW1.txt	sinc	2.194873066612718
NEW1.txt	becom	1.7551882053937472
NEW1.txt	wide	2.1631299484683056
NEW1.txt	us	1.8420892754959475
NEW1.txt	program	2.1631299484683056
NEW1.txt	languag	1.2314936483967598

Fig-7: Calculating Similarity

Fig-8 shows the clustering of objects. Here related words are grouped into a single cluster and the various clusters are displayed.



Ontology Based Text Summarization

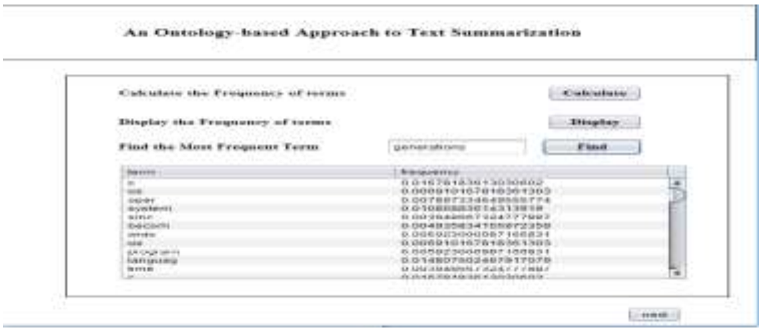
Find the Cluster

Display the Cluster Objects

Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt
Cluster4	NEW2.txt

Fig-8 : Finding Cluster and displaying the Cluster Objects.

Fig-9 shows the calculation of frequency of various terms and finds the most frequent term from the input files.



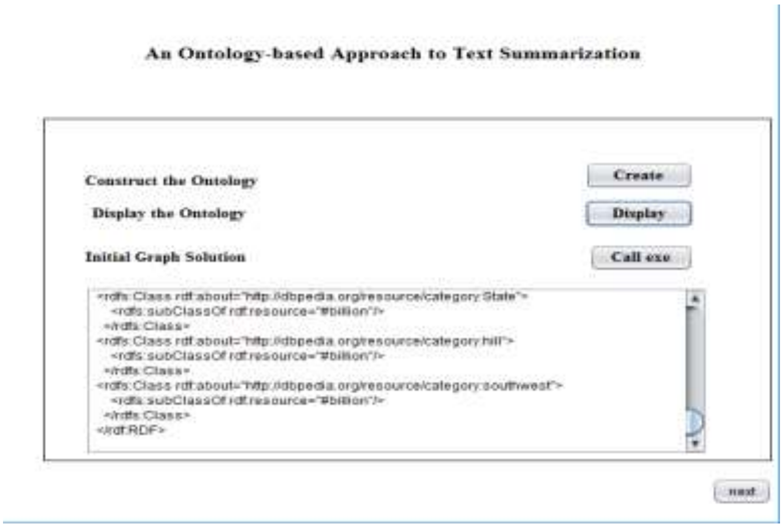
An Ontology-based Approach to Text Summarization
Calculate the Frequency of terms
Calculate
Display the Frequency of terms
Display
Find the Most Frequent Term
Find
next

generations

term	frequency
a	0.016761836130306602
us	0.000910167816361303
paper	0.007887234648555774
system	0.01088883614313818
since	[illegible]
become	0.004835834155872358
ends	[illegible]
we	[illegible]
program	[illegible]
language	0.014807502487817078
time	[illegible]

Fig-9: Calculating Frequency of Terms

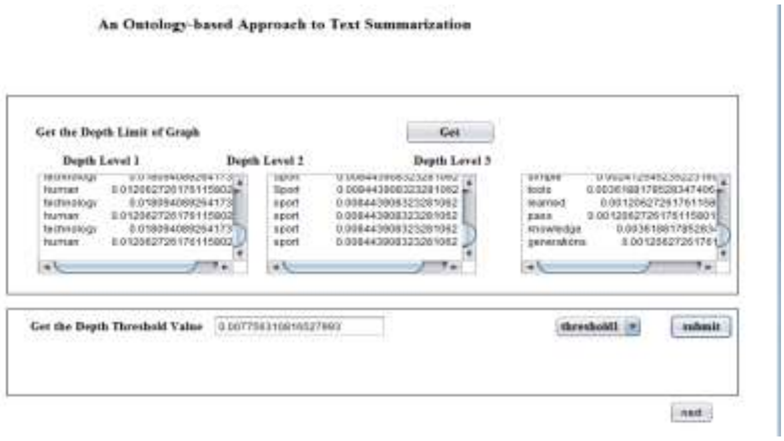
Fig-10 shows the construction of ontology. It takes reference of words from the dictionary and it executes the batch files.



```
<rdfs:Class rdf:about="http://dbpedia.org/resource/category:State">
   <rdfs:subClassOf rdf:resource="#billion"/>
  </rdfs:Class>
<rdfs:Class rdf:about="http://dbpedia.org/resource/category:hill">
   <rdfs:subClassOf rdf:resource="#billion"/>
  </rdfs:Class>
<rdfs:Class rdf:about="http://dbpedia.org/resource/category:southwest">
   <rdfs:subClassOf rdf:resource="#billion"/>
  </rdfs:Class>
</rdf:RDF>
```

Fig-10: Construction of Ontology

Calculation of depth limits and finding the depth threshold values are shown in Fig-11. The depth limit of various levels and the threshold values are shown.



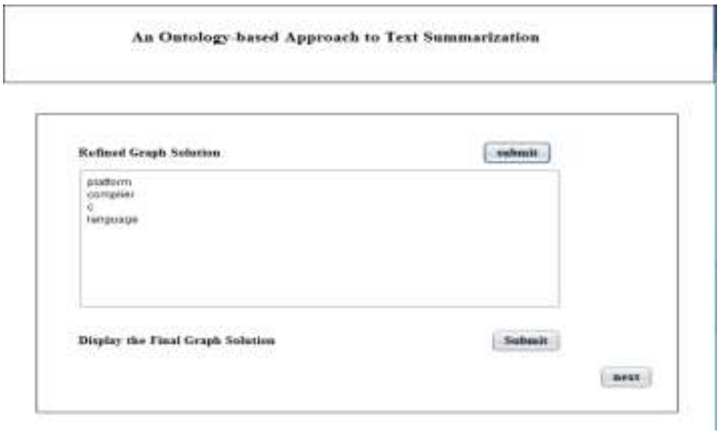
An Ontology-based Approach to Text Summarization
Get the Depth Limit of Graph
Get
threshold1
submit
next

Depth Level 1		Depth Level 2		Depth Level 3	
technology	0.018894088264173	sport	0.008443908323281062	simple	0.0024125452356223
human	0.012062726175115802	Sport	0.008443908323281062	tools	0.0036188178528347406
technology	0.018894088264173	sport	0.008443908323281062	learned	0.0012062726176115
human	0.012062726175115802	sport	0.008443908323281062	pass	0.00120627261751158O1
technology	0.018894088264173	sport	0.008443908323281062	knowledge	0.00361881785283
human	0.012062726175115802	sport	0.008443908323281062	generations	0.00120627261761

Get the Depth Threshold Value 0.007758310816527893

Fig-11: Getting the Depth Limit of Graph

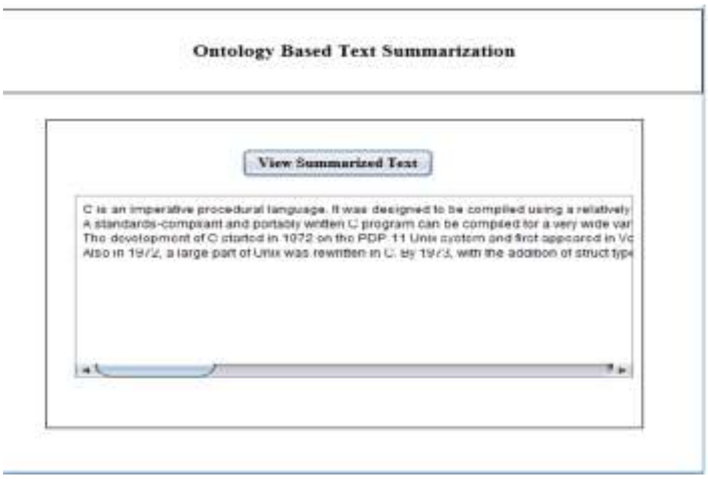
Fig-12 shows the refined graph solution. It takes the words with the highest scores and displays them.


An Ontology based Approach to Text Summarization
Refined Graph Solution
submit
Display the Final Graph Solution
Submit
next

```
platform
compiler
c
language
```

Fig-12: Obtaining the Refined Graph Solution.

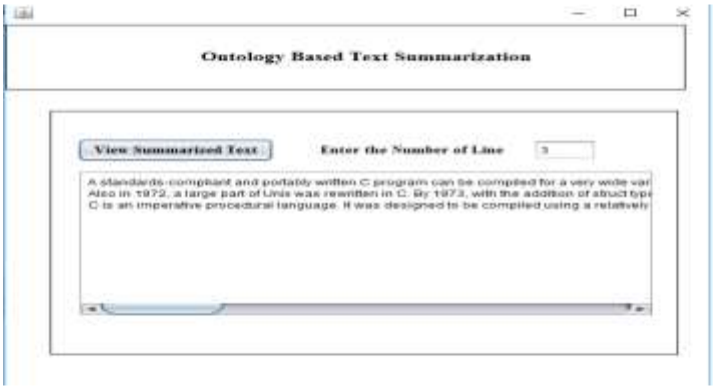
Fig-13 shows the summarized text. Here the condensed version of the input files that is the summary is given by collecting the results of the refined graph solution.



```
C is an imperative procedural language. It was designed to be compiled using a relatively
A standards-compliant and portably written C program can be compiled for a very wide var
The development of C started in 1972 on the PDP 11 Unix system and first appeared in Vc
Also in 1972, a large part of Unix was rewritten in C. By 1973, with the addition of struct type
```

Fig-13: Generation of Final Summary

Fig-14 shows the ranked summary. It gives the final summary in the number of lines as specified by the user.



Enter the Number of Line: 3

```
A standards-compliant and portably written C program can be compiled for a very wide var
Also in 1972, a large part of Unix was rewritten in C. By 1973, with the addition of struct type
C is an imperative procedural language. It was designed to be compiled using a relatively
```

Fig-14: Summarization with specified lines.

CONCLUSIONS AND FUTURE WORK

In this work, the initial document corpus is refined into the form of summarization. In this process, we select only the effective features from the refined graph solution. Obtained results are overall summary of the input document corpus. Summaries provide readers with condensed version of information which is most relevant in the document. It helps readers to assess the document's value without reading the whole document. It acts as content repositories for extracting valuable facts or information. It also gives the final summary in the number of lines as specified by the users. In future this work can be extended to topics focused on summarization framework to news articles or blogs and to also to various machine learning approaches. The new metrics can be investigated which can be used in automatic evaluation environment to measure the overall quality such as grammar, readability, prominence and relativeness. Research in summarization continues to enhance the diversity and information richness and strive to produce coherent and focused answers to users information need.

REFERENCES

1. Varma, M. M. M., & Nandimath, J. (2014). An ontology-based text mining method to construct d-matrix for fault detection and diagnosis using graph comparison algorithm.
2. Zhong, N., Li, Y., & Wu, S. T. (2012). Effective pattern discovery for text mining. *IEEE transactions on knowledge and data engineering*, *24*(1), 30-44.
3. Chen, B., Lam, W., Tsang, I. W., & Wong, T. L. (2013). Discovering low-rank shared concept space for adapting text mining models. *IEEE transactions on pattern analysis and machine intelligence*, *35*(6), 1284-1297.
4. Jiang, J. Y., Liou, R. J., & Lee, S. J. (2011). A fuzzy self-constructing feature clustering algorithm for text classification. *IEEE transactions on knowledge and data engineering*, *23*(3), 335-349.
5. Ma, J., Xu, W., Sun, Y. H., Turban, E., Wang, S., & Liu, O. (2012). An ontology-based text-mining method to cluster proposals for research project selection. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *42*(3), 784-790.
6. Shehata, S., Karray, F., & Kamel, M. (2010). An efficient concept-based mining model for enhancing text clustering. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1360-1371.
7. Zhang, X., Cui, L., & Wang, Y. (2013). Computing Multi-Dimensional Trust by Mining E-Commerce Feedback Comments. *IEEE Transactions on Knowledge & Data Engineering*, (1), 1.