ISSN No. 0976-5697

Volume 1, No. 3, Sept-Oct 2010

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Frequent Term Based Clustering of Stories with Semantic Analysis for Searching and Retrieval

Amrut Nagasunder*, Bharath Boregowda, Madhu Venkatesha, Ananthanarayana V. S.
Dept. of Information Technology
National Institute of Technology, Karnataka
Surathkal, India
{amrut.nagasunder, bharath.boregowda, madhu.venkatesha}@gmail.com, anvs@nitk.ac.in

Abstract: Effective document organizations are often those which provide a concise representation of text content in a large collection of documents. We have considered the task of clustering of stories (documents) as a facilitation of effectual document arrangement for searching and retrieval. We propose a novel representation for a story, based on the essential parts of speech - the nouns, verbs and adjectives. We then perform a clustering of these story representations, resulting in a graph structure where the story representations are conjoined at nodes having the same or synonymous noun. Such a structure can be queried for stories by giving a search string. We employ the use of a knowledge bank throughout the system as a step to realize semantic analysis of the text. For testing the goodness of cluster, we carry out the classification test, on two data-sets. We are able to achieve significantly high quality of clustering, with promising results in regard to memory compaction.

Keywords: Document Clustering, Semantic Analysis, Text Mining, Natural Language Processing

I. INTRODUCTION

Location and extraction of interesting information is one of the hallmark tasks in Information Retrieval (IR). A robust IR system takes a query from a user and responds with the most relevant set of documents. This is done using a document organization approach that facilitates removal of non-relevant documents in the retrieved set. A number of alternative approaches have been proposed to document organization over the years. These approaches focus on visualization and presentation of some relationships among document terms, or the user query [4, 5 and 1]. One such approach is document clustering. In this paper, we propose a novel approach to clustering stories along with an apposite search interface for document retrieval based on a user query. We represent the stories by their frequently occurring nouns, and consider commonalities in features of the stories by bringing similar stories closer together.

One can justify such an approach by arguing that nouns (in specific, proper nouns) form an important source of information in relevant documents' detection and content extraction from text [10]. Thus, assuming that text documents conform to standard grammatical and linguistic constructs, we expect that this representation of documents would provide a compact yet adequate description of the textual content of the documents.

With this work, we present two closely interlinked concepts of IR. First, we explore a novel approach to effective document clustering (which has been recognized as a better method to document organization than traditional ranked lists [9]), mainly established on the basis of Cluster Hypothesis: "Closely associated documents tend to be relevant to the same requests" [3, 6]. Additionally, we propose a search interface designed in such a way that it is apt to our representation and organization of documents. We have incorporated certain heuristic production rules in user query input and backed it by an optimized search strategy that complements our document organization.

In addition to rendering a graph-based text data mining approach, we further furnish our system by providing scaffolding from a linguistic perspective. Previous research shows that WordNet, a lexical dictionary for English language, improves text document clustering [7]. We utilize the hierarchical structure of WordNet for bringing in a sense of semantic analysis in our document organization and retrieval. We also prove that use of WordNet improves the extent of compaction achieved in document organization.

In the remainder of the paper, we describe our working prototype in detail and discuss directions for our continued research.

II. STORY ANALYZER ARCHITECTURE

In this section, we describe the architecture of our system, which is displayed in Fig. 1. Notice that there are 3 main components of the architecture, namely Feature Vector Generator, Cluster Generator and Query Interface. Each document in the data-set is channelled through the Pre-processor and then directed to the Feature Vector Generator, which does the document analysis and representation. The Cluster Generator takes this structured representation of each of the text documents and generates a clustering based on some common terms' matching and merging. The Query Interface is a typical search engine, which retrieves all documents germane to the identified cluster-section, pertaining to the input search query. Throughout the framework of the system, we have employed the use of a knowledge bank, WordNet, to instill an element of semantic analysis. Each of these main components is described in greater depth in the subsequent sections.

A. *Feature Vector Generation*

We present a unique mechanism for document representation. Each document is pre-processed and then analyzed, by extraction of nouns and their associated adjectives and verbs. Each unique noun with its associated adjectives and verbs is considered as a separate node (also referred to as "WordNode"). The nodes of a particular document are linked together in a list which we refer to as the 'Feature Vector' of that document. Each feature vector is truncated based on the frequency of occurrence of nouns, in order to achieve a higher extent of data compaction. Also, the nodes are alphabetically ordered with respect to nouns.

B. *Clustering*

We employ an agglomerative approach, by considering one feature vector at a time, and adding each node of the feature vector iteratively, taking into account each of the corresponding edges to a given node. A directed graph structure is established; the nodes having same nouns are merged and the edges adjusted to bring in compaction to the cluster. Associated adjectives and verbs are not merged but appended to the adjective and verb lists. The feature vectors' node structure is devised in such a way that during graph creation and graph traversal, loops are avoided, and repeated traversal is prevented. Strictly speaking, we would have a forest, not a totally connected graph.

C. *Search*

This is where user interaction would come into picture. The search query entered by the user is first decomposed based on certain defined production rules. The decomposed query is then sampled on the graph, and the relevant documents are retrieved. We also show how our document representation can be utilized in optimizing the search, for refinement of the displayed search results.

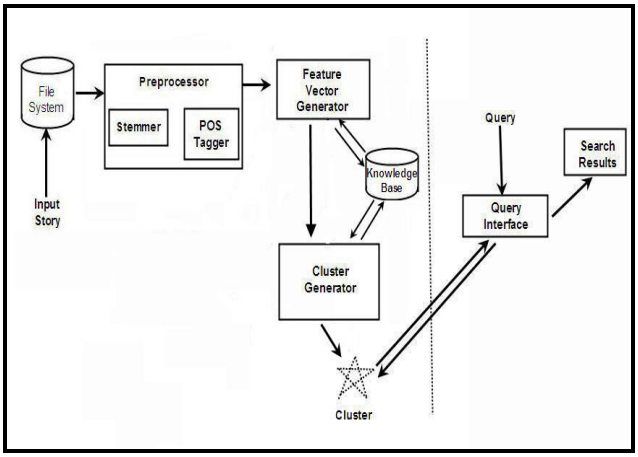

Figure 1. Story Analyzer Architecture

III. LITERATURE SURVEY

Reference [8] provides a hierarchical 'monothetic' document clustering algorithm for summarization and browsing search results. The authors state that 'Monothetic algorithms are those in which documents are assigned to a cluster based on a single characteristic feature'. Their algorithm progressively identifies topics in a way that maximizes the *coverage* while maintaining *distinctiveness* of the topics. In their clustering algorithm, they assume that each document in the collection can be represented by a set of concepts. Each node in the hierarchy is associated with a concept (the node label) and all documents under that node contain that concept. In general, each of these documents will contain several other concepts as well. They refer to the union of all these concepts as *concepts under the node*. Monothetic clustering of the documents under a node involves selecting a subset of those concepts, optimal in some sense, and associating a child node with each of them.

In the paper titled "Frequent term-based text clustering" [2], the authors introduce an approach which uses frequent item or term sets, discovered using association mining rules, to represent the documents. To cluster the data they measure the mutual overlap of frequent item sets of the supporting documents. They employ a greedy approach in generating two clustering algorithms: a flat clustering and a hierarchical clustering. Their hierarchical clustering generates *overlapping* clusters, using frequent k-term sets at the k-th level.

Our system employs a unique document representation that is frequent-noun centric and a graph-based technique in document organization, with additional support from WordNet knowledge bank, which, to our best knowledge, has never been attempted thus far.

IV. DATA PRE-PROCESSING

This is the first phase of the system. Pre-processing involves stop-word removal, stemming, lower-casing and part of speech (POS) tagging of words. Stop-word removal involves eliminating commonly occurring terms such as "a", "the" etc. Stemming is bringing each word to its root form.

For the POS tagging, we used the Stanford POS Tagger [11].

V. FEATURE VECTOR GENERATION

The purpose of this stage is to obtain a compact yet adequate representation for each story, by taking only certain essential elements of the story, as described below.

We consider nouns, verbs and adjectives as the essential parts of speech of a story in the context of a good representation for it. Of these, the nouns are the most essential parts of speech of a story. We propose the argument that, a story revolves around nouns; every adjective and verb in each sentence is associated with some noun in it, and adverb describes a verb. Hence the nouns form a special object of interest in our system.

All the nouns, with associated adjectives and verbs are extracted. Other parts of speech are removed. This association is determined by co-occurrence of the terms within the same sentence. Each of these noun-adjectives-verbs groups is a potential member in the representation of a story. For greater succinctness, only the more frequent nouns are selected, as described later in this section.

After extraction of the noun-adjectives-verbs groups, each noun's frequency count, which is the number of times it occurs in the story, is calculated. For each counted noun, a check is done with the synonyms of that noun. If two nouns are synonyms or hyponyms of each other, they are merged into a single, common node, with a frequency count as the sum of their individual counts.

A. *Node Structure*

Fig. 2 shows the structure of a node. The field 'Edges' is used to point to the next node in the story representation. The same field may point to more than one node in the cluster graph, as we shall see later. That is why the 'Edges' field is shown with more than one arrow. For generation of synonym/hyponym set of each word, we use WordNet.

Each node also contains a mapping table called 'HashMap' Table, which contains a list of Edge-to-Edge mappings, accounting every incoming edge to its corresponding outgoing edge (Note that multiple edges may enter in and exit from a node). This HashMap Table is crucial in graph traversal, in avoiding loops, preventing repeated traversal and forbidding wrong story retrieval.

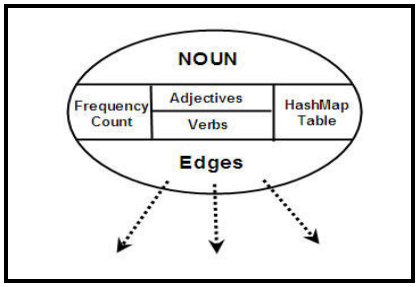

Figure 2. Node Structure

B. *Frequency Statistics*

In computing the frequency statistics of a given story, we set a minimum threshold on frequency of occurrence of the words, and the most frequently occurring terms are selected, thereby truncating insignificant parts in the representation.

For ensuring equal representation of all documents (note that documents may be of different lengths), we have different threshold values in computing the frequency statistics. Different "slabs" are set based on the varying lengths of the documents. The slabs for the Stories data-set are set (empirically) as shown in Table I. A threshold of 1 means all the WordNodes with noun count of 1 or more will be selected. As an example, the frequency statistics for the story "The Wolf and the Crane" (See APPENDIX) is shown below in Table II. The number of nouns in this story is ten (<15). So, all of them are taken for story representation.

Table I. Slabs for Frequency Threshold

Number of Terms	*Threshold*
15	1
50	2
100	3
150	4
200	5

Table II. Frequency Statistics for "The Wolf and the Crane"

Word	*Frequency Count*
Bone	3
Crane	3
Head	2
Jaw	1
Mouth	2
Pain (Injury)	2
Sum (Payment, Reward, Recompense)	4
Throat	1
Tooth	1
Wolf	3

Note that, the relationship between injury and pain is not *synonymy* but *hyponymy*. Injury (hyponym) is a pain (hypernym), but a pain need not be injury. The representative noun will be the hypernym. Similarly, payment, recompense and reward (hyponyms) are all sums (hypernym). Hence they are merged into the same node, with a frequency count of 4. Even the associated adjectives and verbs will be merged.

C. *Generation of Feature Vector*

A 'feature vector' is the actual representation for a story. It contains the essential features – those noun-adjectives-verbs groups which pass the frequency threshold test – as a list of nodes arranged alphabetically with respect to nouns

and connected through the 'Edges' field. The last (leaf) node of a feature vector points to the story it represents. Fig. 3 shows the feature vector for the story "The Wolf and the Crane".

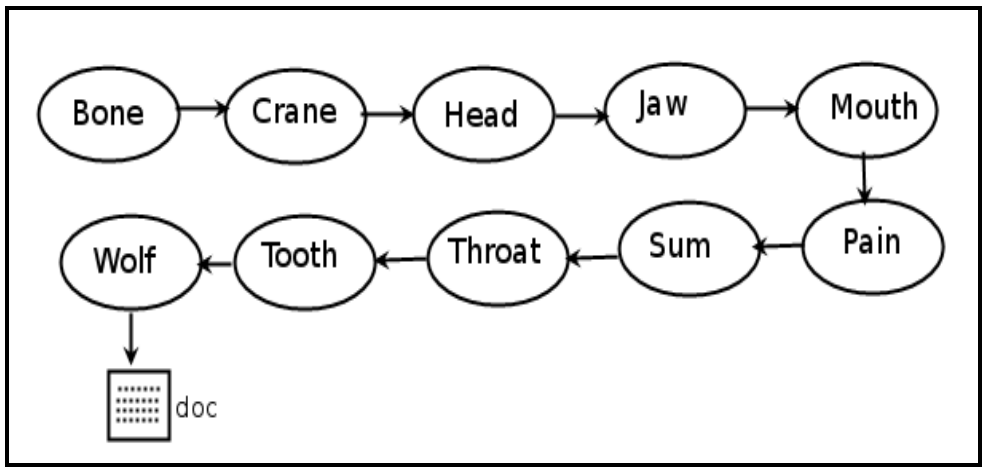

Figure 3. Feature vector for "The Wolf and The Crane"

VI. CLUSTERING

The next phase in the system is to cluster the stories such that similar stories, with respect to certain features, are together. In our system we use the noun similarity in clustering. The clustering will result in a graph structure, with nodes with same or synonymous nouns joining the different feature vectors.

A. Creation of Graph Structure

A directed graph is created by taking one feature vector at a time. Nodes of the feature vector are iteratively added to the existing graph. A list of '**Root**' nodes is maintained; these nodes are the points of entry into the graph while searching or building the graph. All the nodes in this list will be the '**Head Node**' of some feature vector. If the noun in the Head Node of a feature vector or its synonyms is not present in the graph, then it is added to the '**Root-list'**. Else, it is merged with the matching node and the edges will be updated.

Addition of any node of the same feature vector, henceforth "FV", apart from the Head Node and its immediate successor, results in HashMap table update. At each stage, after the Head Node and its successor have been added, till end of FV, once a particular node has been added or merged in the graph, an entry is added in the HashMap table of the immediate previously added node, indicating the corresponding 'From' and 'To' edge pair. A variable called '**Current-Node**' is used for tracking purposes. In merging, adjectives and verbs of the Current-Node are appended to the adjective and verb lists of the already existing node. Table III gives the algorithm 'Create-Graph' used to construct the cluster graph. Fig. 4 is a depiction of a typical merged node. Adj1 and Vb1 stand for adjective list and verb list of node 1, Adj2 and Vb2 stand for adjective list and verb list of node 2 and so on.

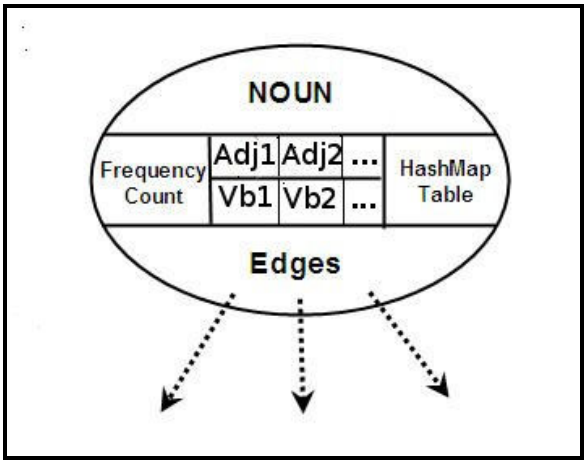

Figure 4. Merged Node

Figure 5 displays the graph structure for the following stories:

- The Wolf and the Crane
  (Bone, wolf, crane, sum, head, mouth)
- The Wolf and the Lamb
  (Lamb, wolf, tyrant)
- The Wolf in the Sheep's Clothing
  (Wolf, lamb, meal, skin)
- The Boy Who Cried Wolf
  (Sheep wolf and shepherd)

In Fig. 5, the graph structure shows ordering the nodes of the feature vectors in the order of decreasing frequency count of the nouns and not alphabetical order for better visualization. Also, only a few of the more frequent nouns are considered for the same reason.

Table III. Algorithm Create-Graph

```
Begin:
    Initialize Root, Current-Node.
    For each document-FV [i] in document-list
    Begin:
        For each node N in document-FV[i]
        Begin:
            If Noun(N) NOT present in Graph,
              For each Synonym in Synonyms (Noun(N))
               Check for presence of Synonym in Graph.

              If no Synonym is present in Graph,
                If N is Head Node of document-FV[i],
                  Append N to Root-list
                Else,
                  Add New Node = Successor of Current-Node
                  Update HashMap Entries

            If Noun (N) Or Synonym (Noun(N)) is Present in Graph,
              Merge Current-Node with Already Existing Node.
              Update HashMap Entries.

            Update Current-Node.
        End
    End
End
```

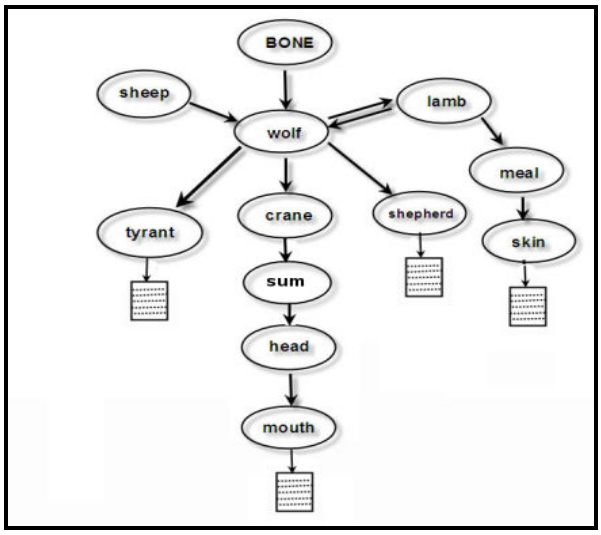

Figure 5. Sample Cluster Component

VII. SEARCH INTERFACE

Searching for the stories will be done depending on the search terms. The search term or the query is first decomposed and then taken up for search.

A. *Query Decomposition*

Before decomposing the query, the search terms are POS-tagged. For suitability, we have defined certain rules for a valid query string. The rules are generic, and formulated in a way that make them apt for our system. We define these

rules as a set of production rules as shown in Table IV. We expect the user to issue queries confirming to these rules only, else the query will be treated as invalid.

Table IV. Valid Query Production Rules

Rule 1:	Query **:=** Query OR Query | ε
Rule 2:	Query **:=** Group Query
Rule 3:	Group **:=** P1 noun P2
Rule 4:	P1 **:=** P1 adjective | ε
Rule 5:	P2 **:=** P2 verb | ε

Following the above rules, 'adjective1 noun1 verb1 noun2 noun3 verb2' is a valid query, where as 'adjective1 verb1 noun1' is invalid. In the former, there is one query with three groups – (adjective1 noun1 verb1), (noun2), (noun3 verb2). Notice that there is one and only one noun in every group, and adjective(s) and verb are associated with it. In the latter, association is not proper. A search string like 'cunning fox waiting crow sparrow perching' – will be decomposed into three groups like {cunning(adj), fox(n), waiting(vb)}, {crow(n)}, {sparrow(n), perching(vb)}. Also the search terms are considered alphabetically, with respect to nouns, while searching – so the order of searching will be crow, fox and sparrow.

B. Search Strategy

The nouns of the decomposed query are then taken up as separate query terms. The sequence of query terms is taken as the "window" to be searched in the graph. In the event that the window returns no results, windows of lower size are progressively explored, and the search is conducted. Alternatively, if a query doesn't return any documents, the search is re-run, this time with the synonyms of the query terms. Every term's synonyms are considered, and the search is conducted.

Each term is searched through the graph, and the "matching" nodes are obtained. From the first matching node, the edges leading from that node are traversed, and the process is repeated recursively until all the query words of the window are explored.

When adjectives are present in the search term, then first the noun is searched. If there is match, then in that node the adjectives list is searched for the presence of those or synonymous adjectives. If there is match, then the edge corresponding to that member node of the adjective list is noted for further traversal. Similarly, when verbs are present, verbs list is searched for match. When both present, only match in both the lists ensures further traversal down the corresponding edge.

C. Search Optimization

We further optimize our search as follows: Once the matching node of the first query term is obtained, we search for the second query term starting with this node, and from the second query node, we utilize the HashMap information of that particular node in order to continue the traversal till the node of the last query term. The documents are retrieved, using the HashMap, traversing down the last query term node. As a result, the paths that connect all query terms in that window sequence is obtained, and the set of documents for that query are displayed.

VIII. PERFORMANCE EVALUATION

A. Goodness of Cluster

For evaluating the goodness of cluster, we employ a quantitative mechanism. A standard test for goodness of clustering is percentage of correctly classified test instances on pre-labelled data. The data-set is randomly divided into Training Data and Test Data with about 80% in Training Set. The clustering is then done with the Training set. The FVs of the stories in the Testing set are then sampled through the graph for 'best fit'.

For finding the best fit, a varying length window of the FV is run through the graph, and largest sized windows that match with the graph are considered. The stories corresponding to these windows are retrieved, and the majority label of the stories in this set is applied to the test story. Note that the test story will have its pre-test label. If this is same as the label applied, then there is a match, which is used in measuring the goodness of cluster.

As a metric of system performance evaluation, the Accuracy Measure (AM) is considered as:

$$AM = \frac{n \text{ (Test stories having matching class label)}}{n \text{ (Test Stories)}} \times 100 \quad (1)$$

B. Memory Compaction

We also measure the extent to which our cluster achieves compaction in data representation. A metric to evaluate the degree or "extent" of compaction (EC) is conceived as:

$$EC = \frac{\text{Memory (FVs)} - \text{Memory (cluster)}}{\text{Memory (FVs)}} \times 100 \quad (2)$$

Memory ( ) represents the memory usage by the parameter within the parenthesis. Note that the value of memory usage for FVs is used, and not that of the original document set. This is because, after the pre-processing stage, most of the textual content is removed, and the essential features are captured in the FV. The clustering is then done with the FVs, and not the original documents.

A high value of Extent of Compaction shows greater compaction in data, and hence is desirable.

IX. RESULTS

We tested our system on two data-sets:

- *Short Stories*: Short stories were collected from various sources (See APPENDIX) and compiled into one data-set. These stories were manually tagged with labels, based on story content and source of collection. About 125 short stories were collected for this.

- *Reuters Data Set*: This data-set consists of 21578 articles from the Reuters news service in the year 1987 (http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html). A subset consisting of 4744 documents is used for evaluation.

A. *Goodness of Cluster*

1) Short Stories

Training Data: 101
Test Data: 24

Classes: 5

- Alladin
- Birbal
- Buddha
- Hodja
- Ramalinga

Table V. Goodness of Cluster (Short Stories)

Sl. No.	*Story Name*	*Alladin*	*Birbal*	*Buddha*	*Hodja*	*Ramalinga*	*Match*
1	Alladin -- The Story of the Blind Baba-Abdalla	4	0	0	0	0	TRUE
2	Alladin -- The Story of the Merchant and the Genie	15	11	7	5	1	TRUE
3	Alladin -- The Story of the Second Calender, Son of a	14	11	5	9	0	TRUE
4	Alladin -- The Story of the Young King of the Black Isles	15	12	6	6	6	TRUE
5	Birbal -- Back to Square One	11	20	6	6	8	TRUE
6	Birbal -- Birbal Outwits Cheat	0	15	0	2	2	TRUE
7	Birbal -- Birbal Shortens Road	0	22	1	1	2	TRUE
8	Birbal -- Limping Horse	1	1	2	2	0	FALSE
9	Birbal -- The Sharpest Sword and Shield	7	13	0	8	2	TRUE
10	Buddha -- Nalgiri Elephant	1	0	14	0	0	TRUE
11	Buddha -- Story of Kumara Kassapa	7	8	19	2	8	TRUE
12	Buddha -- Story of Mara	0	0	19	0	0	TRUE
13	Buddha -- The Buddha's Victory over Mara	0	0	19	0	0	TRUE
14	Buddha -- The Four Sights	8	6	19	4	1	TRUE
15	Buddha -- The Savatthi Miracles	0	0	20	0	0	TRUE
16	Hodja -- Rich Dream	8	10	2	17	3	TRUE
17	Hodja -- Sour Reply	12	12	4	19	2	TRUE
18	Hodja -- Sweet Quarrels	6	3	1	19	2	TRUE
19	Hodja -- The Mulla in Muddle	3	0	0	19	1	TRUE
20	Hodja -- The Relatives of Donkey	7	6	0	19	0	TRUE
21	Ramalinga -- Mahabharat and Delhi Sultan's wish	9	8	2	3	10	TRUE
22	Ramalinga -- Ramalinga's Entry Into Bhuvana Vijayam	7	10	2	2	12	TRUE
23	Ramalinga -- Thathacharya - Demon Chanting Hymns	13	13	3	9	12	FALSE
24	Ramalinga -- The Secret of Weaving Invisible Fabric	7	11	2	2	12	TRUE

The test results for all the 24 stories are listed in Table V. Consider Story No. 16 ‘Hodja – Rich Dream’, for example. When this story (it’s FV) was sampled on the graph, it retrieved 8 ‘Alladin’ stories, 10 ‘Birbal’ Stories, 2 ‘Buddha’ stories, 17 other ‘Hodja’ stories and 3 ‘Ramalinga’ stories.

So its applied label is ‘Hodja’, which is same as its pre-test label. Hence the match is ‘TRUE’.

Accuracy **=** 22/24 **= 91.67 %**

Hence we can say that ‘**Confidence**’ in correct prediction of class label for stories **= 0.917.**

2) Reuters News Stories

Training Data: 4744
Test Data: 280

The data in this corpus have been classified into different categories, with most documents having multiple classes (topic, people, location etc). We have considered for our analysis, only those documents belonging to the following classes, based on location.

Classes: 8

- France
- Germany
- Brazil
- Japan
- Iran
- China
- Canada

- Italy

Table VI shows the classification statistics for a few documents belonging to the test set of Reuters text corpus. Table VII shows the statistics for the entire test set comprising of entire set of documents.

Table VI. Goodness of Cluster (Reuters) - 1

Sl. No.	*Document*	*France*	*Germany*	*Brazil*	*Japan*	*Iran*	*china*	*Canada*	*Italy*	*Match*
1	1-APR-1987 France	167	60	44	135	2	19	37	28	TRUE
2	1-JUN-1987 France	275	133	33	147	2	18	107	26	TRUE
3	7-APR-1987 Iran	2	2	0	0	18	0	0	0	TRUE
4	9-APR-1987 Brazil	3	5	208	12	1	0	2	3	TRUE
5	9-APR-1987 Japan	40	53	18	503	5	15	34	9	TRUE
6	9-MAR-1987 Brazil	0	9	197	8	16	2	12	2	TRUE
7	9-MAR-1987 China	6	9	12	22	28	141	39	1	TRUE
8	9-MAR-1987 Germany	125	273	108	290	4	45	168	54	FALSE
9	9-MAR-1987 Canada	13	13	53	47	7	18	819	6	TRUE
10	9-MAR-1987 Canada	25	18	5	43	0	2	830	11	TRUE
11	9-MAR-1987 Japan	112	161	35	494	9	34	147	34	TRUE

Table VII. Goodness of Cluster (Reuters) - 2

Class	*Number of Test Cases*	*Correctly Predicted*	*Wrongly Predicted*
France	30	22	8
Germany	40	22	18
Brazil	30	24	6
Japan	70	68	2
Iran	8	6	2
China	20	13	7
Canada	70	69	1
Italy	12	5	7
Total	**280**	**229**	**51**

Accuracy = 229 / 280 = **81.78 %.** Confidence **= 0.818** .

B. Memory Compaction

Table VIII gives the memory compaction details for the Short Stories and Reuters data – with and without WordNet. The statistics show a good extent of compaction in all four cases. Since the number of documents in the Short Stories data-set is less, a lower value for extent of compaction, as seen, can be expected.

Table VIII. Memory Statistics

	No. of Documents	*Size on disk of Original docs*	*Size of Feature Vectors*	*Size of Graph Structure*	*Compaction*
Short Stories	101	437 KB	208.5 KB	79.8 KB	61.73 %
Reuters	4744	19.2 MB	6.97 MB	1.55 MB	77.76 %
Short Stories Without WordNet	101	437 KB	208.5 KB	93.7 KB	55.06 %
Reuters Without WordNet	4744	19.2 MB	7.80 MB	2.19 MB	71.92 %

X. CONCLUSION AND FUTURE DIRECTIONS

In this paper we have described a working prototype system for effective document clustering based on frequently occurring terms with scaffolding from a linguistic perspective. This research work has provoked us with two stimulating questions worth pursuing:

- Splay Tree Concept in Searching: Could the graph be made to "re-arrange" itself dynamically so as to ensure that more frequently searched terms tend to occur at the "top" of the graph (forest), to reduce search times?

- Memory Usage and Data Storage: If the cluster size is too large, the system might run out of memory. Hence, it becomes vital to ensure that adequate memory is available during the system run. A step towards this end is to store independent components of the forest on the disk, and retrieving those components as and when required, appropriately.

We have developed a novel approach to document clustering using a graph-based technique, with assistance of a knowledge bank, and tested it on multiple text corpora. Our results are promising, suggesting that further research may result in much better performance and possibly even direct commercial applicability.

XI. REFERENCES

[1] Allan, J., 1997. "Building hypertext using information retrieval," Information Processing and Management, 33(2), pp. 145 -159.

[2] Beil, F., Ester, M., Xu, X., "Frequent term-based text clustering," Proceedings of the 8th ACM SIGKDD international conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, 2002.

[3] Croft, W. B., "Organizing and searching large files of documents," PhD thesis, University of Cambridge, 1978.

[4] Croft, W. B., Thompson, R. H., "I^3R: A new approach to the design of document retrieval systems," Journal of the American Society for Information Science, 38: pp. 389-404, 1987.

[5] Dubin, D., "Document analysis for visualization," In Proceedings of ACM SIGIR., 1995.

[6] Hearst, M. A., Pedersen, J. O, "Re-examining the cluster hypothesis: Scatter/Gather on retrieval results," In Proceedings of ACM SIGIR, pp. 76-84, 1996

[7] Hotho, A., Staab, S., Stumme, G., "WordNet improves text document clustering," In Proceedings of the Semantic Web Workshop at SIGIR, 26th Annual International ACM SIGIR Conference, Toronto, Canada, 2003.

[8] Kummamuru, K., Lotlikar, R., Roy, S., Singal, K., Krishnapuram, R., "A hierarchical monothetic document clustering algorithm for summarization and browsing search results," Proceedings of the 13th international conference on World Wide Web, New York, NY, USA, 2004.

[9] Leuski, A., "Evaluating document clustering for interactive information retrieval," Proceedings of the 10th international conference on Information and knowledge management, Atlanta, Georgia, USA, 2001.

[10] Rau L., "Extracting company names from text," Proceedings of the 7th Conference on Artificial Intelligence Applications. Miami Beach, Florida, 1991.

[11] Toutanova, K., Manning, C.D., "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger," In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000), pp. 63-70, 2000.

XII. APPENDIX

A. *The Wolf and the Crane*

A Wolf who had a bone stuck in his throat hired a Crane, for a large sum, to put her head into his mouth and draw out the bone. When the Crane had extracted the bone and demanded the promised payment, the Wolf, grinning and grinding his teeth, exclaimed: "Why, you have surely already had a sufficient recompense, in having been permitted to draw out your head in safety from the mouth and jaws of a wolf." In serving the wicked, expect no reward, and be thankful if you escape injury for your pains.

B. *Sources of Short Stories*

- http://www.eastoftheweb.com
- http://www.indianchild.com
- http://www.gutenberg.org