Document Modeling and Clustering using Hypergraph

Dhanya P.M*, Sreekumar A, Jathavedan M
Department of Computer Applications,
Cochin University of Science and Technology,
Kochi, Kerala, India.
*ORCID: 0000-0002-2988-8782

Ramkumar P.B
Department of Basic Sciences and Humanities,
Rajagiri School of Engineering and Technology,
Kochi, Kerala, India.

Abstract
Documents have been represented using graphs for many applications like document clustering, document summarization etc. They have also been modeled using the vector space model for various text processing activities. The purpose of this paper is to model text using hypergraph and apply the morphological operator on hypergraph created from the underlying text to get text clusters. The document is considered as a graph and partitioning is applied which finally results in clustering. Here document is modeled as a hypergraph and two methods for text clustering are discussed. The first method uses simple hypergraph and the second method uses a weighted hypergraph. The paper also discusses on how to model multiple documents as hypergraph. The method can be extended for multidocument clustering also.

Keywords: hypergraph; hypernode; hyperedge; clustering;

INTRODUCTION
Document clustering has been widely used in many information retrieval systems [1]. Clustering helps in finding the nearest neighbour of a document. They are used in search engines in response to a user's query. They also help in creating a hierarchical cluster of documents. Hypergraph is a graph G = (V,E) where V = set of hypernodes and E = set of hyperedges. In paper [2], a hypergraph is modeled for documents where the hypernodes are the documents and hyperedges are the authors. Based on the authors the documents are being grouped. But in our method, while converting to the area of text, a hyperedge is a sentence and hypernodes are the unique words in that sentence. The number of hyperedges in this graph will be the number of sentences considered for clustering. Just as a sentence can have many words in it, a hyperedge is having many hypernodes in it. While modeling multiple documents , a hyperedge can be a document itself and the hypernodes in it can be the unique words in that document. The number of hyperedges will be same as the number of documents considered for clustering. This is the pioneer work which uses the concept of hypergraph in text clustering as well as document clustering. Consider the input text given below.

“The method proposed here uses a suffix replacement methodology where it creates a Malayalam dictionary of suffixes and replacements. The Malayalam suffix replacement dictionary is being developed using MYSQL. In this method tree is being constructed from a set of rules available in the database. The tree of rules is being pickled using the pickling technique of python which makes tree a permanent structure.”

Figure 1: Hypergraph

Modeling of the above text as hypergraph is shown in Fig. 1. Our main aim is to model text using hypergraphs, partition the hypergraph and create text clusters. Outline of the remaining sections of the paper is as follows. Section II presents a detailed literature review on the graphical methods, non graphical methods and parallel algorithms in text clustering. Section III creates a mathematical framework for the text hypergraph and the text weighted hypergraph. Section IV introduces various mathematical operations on the hypergraph and the weighted hypergraph. Hypergraph morphology is discussed in section V. The proposed methodology is discussed in section VI, its implementation, data set and the result analysis in section VII. Section VIII offers some idea on the future work.

LITERATURE SURVEY
This section gives an overview of various methods currently used in the field of text clustering. This paper has broadly divided the text clustering algorithms in to graphical, non graphical methods and its extensions for parallelism. The
graphical methods used are dependency graph based, document graph based, k-NN graph based, co-occurrence graph based, semantic graph based and graphical network of documents. The non graphical methods include k-means, k-medoid, density based, hierarchical methods etc.

A. Non graphical methods

In paper [3], the authors find correlation between the terms using internal correlation measures like association between the terms, normalized association between the terms, covariance and pearson correlation coefficients. Finally these are applied in k-means algorithm to improve the performance. Word sense disambiguation is done using WordNet in [4]. The theme of the text is represented using lexical chains. Only core semantic terms are considered to reduce dimensionality of feature set. The raw input text is subjected to many operations like wide one dimensional convolution, folding, dynamic k-max pooling etc in [5]. The main problem with many algorithms are that they cluster together only if there are common terms. They won't cluster based on concepts. In order to solve this problem, along with document term matrix, document concept matrix [6] is also created with the help of wikipedia and then clustering is done.

A hierarchy of clusters is created in [7] using frequent itemsets. The method spans through several phases like tree construction, pruning, sibling merging etc. Sibling merging is done by finding inter cluster similarity. The method discussed in [8] is about hierarchical clustering. Birch [9] is a bottom up method of clustering. When applied to the document clustering, the CF feature is created from the vector representation of the document and the CF tree created by storing the CF features incrementally. The branching factor is decided and tree nodes are split accordingly. In method [10], density based methods like DBSCAN, CHAMELEON are being discussed, which mainly looks for the density of the key words occurring in the short text. They accumulate neighbours in a dense region and form clusters.

B. Graph oriented methods

The importance of fidel vector in graph partitioning is well demonstrated in paper [11]. Uniform hypergraph partitioning is applied in geometric grouping in [12]. A detailed description on eigen values, eigen vectors and eigen spaces are given in [13]. Eigen value methods are not only used in document processing but also in other fields like image processing [14]. Modeling text using bipartite graph and its clustering are mentioned in [15]. A detailed survey on document representations, text classification, clustering and implementations are discussed in [16].

The method [17] represent sentences and documents as dependency graph. Dependency graph of a sentence shows how words in it are related to each other. Every word in a sentence will be related to the sentence head. After construction of the graph several operations like merging, union etc of edges are performed which would improve the result of clustering. Dependency graph for the entire text can be created. Similarity of the graphs are found out and k-means clustering is applied. A document graph is constructed using WordNet noun taxonomy in [18]. Graphs with similar subgraphs are clustered together. Always similar subgraph reflects similar sense. They are mined using Apriori algorithm. Firstly, they find 1-edge subgraphs, then 2-edge subgraphs, k-edge subgraphs and so on. Finally hierarchical clustering algorithm is applied to find the similar subgraphs.

In method [19], weighted K-NN graph is constructed by assigning each node with k random neighbours with the help of similarity matrix created. In the reduce phase, top-k similar nodes are selected. Edges between the vertices are weighted, where the weight shows the similarity. Edge pruning is done where those below a threshold weight are deleted.

In co-occurrence graph based method, the feature terms in the document will be modeled as vertices and the edges represents the co-occurrence of the terms [20] in the same sentence/paragraph/window of size n. Graphs are subjected to various operations like graph union, subgraph calculation etc. Similar subgraphs are found out and clustering is done.

A network of documents created in [21] with documents as vertices, edges represent similarity between the documents. Now cluster centroids are calculated with the centrality values where the k-means clustering is later applied. Both weighted graph and balanced graph partitioning methods are discussed in [22].

C. Parallel Algorithms

In [23], the authors proposed a parallel k-means algorithm. They create neighbour matrix and include parallelism by including more processors in computing. The initial centroids are updated. They finally show how speed in clustering is improved by increasing the number of processors used. A parallel method with mapreduce is implemented by [24]. The method shows a good scalability and works well on large data sets. The method [25], discusses on hierarchical agglomerative clustering made parallelly. There is increase in the speed and quality of clustering. The method detailed in [26] defines anchors, pivots, hierarchy of anchors and sorting for parallel document processing.

MATHEMATICAL FRAMEWORK FOR TEXT HYPERGRAPH

In this paper, text is modeled using a hypergraph and a weighted hypergraph. A sample hypergraph created is shown in the Fig.2. In the figure there are 27 hypernodes and 3 hyperedges. The second hyperedge is overlapping on the first hyperedge as the nodes 3 and 11 are repeated in second hyperedge also. The third hyperedge is not overlapping on the other hyperedges. While partitioning using the spectral partition method we should get the first two edges in one partition and the third edge in the second partition. So let us describe a mathematical frame work for this hypergraph.

Let \( \Gamma \) denotes the text to be clustered. Let \( \{ H_T, S, \xi, \upsilon \} \) denotes the hypergraph structure corresponding to the text \( \Gamma \). where \( H_T \rightarrow \xi \) is the hypergraph corresponding to the text \( \Gamma, \xi \rightarrow \upsilon \) is the edge set in \( H_T \) which represents the sentences \( S \) in the text \( \Gamma \). \( \upsilon \rightarrow \) the node set in the \( H_T \) which represents the unique words in the text \( \Gamma \). The Edge set \( \xi \) can be partitioned in to disjoint equivalence classes. Equivalence classes generates unique partitions of the given text \( \Gamma \). The same is shown in Fig.2 where the hyperedge set = \{ [1], [2], [3] \} and the hypernode set consists of nodes [1, 2, 3, 4,
Let \((H_w, S, W, \xi, \nu)\) denotes a weighted hypergraph structure corresponding to the text \(\Gamma\) where \(H_w\) the weighted hypergraph corresponding to the text \(\Gamma\), \(\xi\) → the edge set in \(H_w\) which represents the sentences \(S\) in the text \(\Gamma\), \(\nu\) → the node set in the \(H_w\) which represents the unique words in the text \(\Gamma\). The \(W\) → the term frequency of the words in the document. The \(\xi\) can be partitioned in to disjoint equivalence classes. Equivalence classes generates unique partitions of the given text \(\Gamma\).

**A. Lemma 1**

The intersections of the partitions \(P_i\) of the hypergraph \(H_{\Gamma}\) gives an empty set. ie, \(\forall i \cap P_i = \emptyset\).

**Proof**

Let \(P_i\) be the partitions created for edge set \(\xi\). Since the text \(\Gamma\) under consideration belongs to different topics, the edge set can be divided in to different partitions. As in Fig.3, the intersection of these partitions \(P_i\) gives an empty set \(\emptyset\). This is due to the more intra cluster similarity within the partitions and the less inter cluster similarity between the partitions.

**B. Lemma 2**

The union of all the partitions \(P_i\) of the hypergraph \(H_{\Gamma}\) gives the original hypergraph \(H_{\Gamma}\). ie, \(\forall i \cup P_i = H_{\Gamma}\).

**Proof**

The various partitions \(P_i\) created by the method consist of text discussing on various topics. There are also partitions containing outliers . Outliers are text which do not belong to any of the clusters. All these \(P_i\) when joined together will derive the entire original text \(\Gamma\).

**C. Theorem**

Edge set \(\xi\) in the hypergraph \(H_{\Gamma}\) corresponding to the text \(\Gamma\) generates unique partitions \(P_i\) of the text \(\Gamma\). 

**Proof**

An edge in \(H_{\Gamma}\) represents a sentence in \(\Gamma\). So edge set of the hypergraph \(H_{\Gamma}\) represents sentences in the text \(\Gamma\). When spectral partitioning is applied to \(H_{\Gamma}\), edge set is getting divided in to two subsets . These two sets represent the first level clustering of the text \(\Gamma\). When this is repeated iteratively, it ultimately leads to partitions(clusters) which are unique as said in lemma 2 and the intersection of these partitions will be a null set as said in lemma 1.

**MATHEMATICAL OPERATIONS**

The hypergraph \(H_{\Gamma}\) and the weighted hypergraph \(H_{w,\Gamma}\) created for the text \(\Gamma\) undergo various mathematical operations.

**A. Adjacency matrix of \(H_{\Gamma}\)**

The adjacency matrix \(A\) of the hypergraph \(H_{\Gamma}\) is the square matrix of the nodes \(\nu\) in the hypergraph \(H_{\Gamma}\). Here we can see that \(A_{ij} = 1\) and \(A_{ji} = 1\), if the node \(\nu_i\) and node \(\nu_j\) are part of the same hyperedge \(\xi\). Inturn it means both the nodes \(\nu_i\) and \(\nu_j\) are words co-occurring in many sentences. A row in the matrix \(A\) shows the neighboring words of a particular word, taking in to consideration all the sentences in the text \(\Gamma\). Referring to Fig.2, we can see words 1 to 13 and 14 to 18 are neighbours of word 3.

**Figure 2:** Hypergraph for the sample text

**Figure 3:** Partitions of a hypergraph
B. Diagonal matrix of $H_{\Gamma}$

Diagonal matrix $D_{v}$ of the hypergraph $H_{\Gamma}$ is a square matrix of node degree, where the diagonal entries $D_{v_{i}i} = \text{No of hyperedges to which that node } v_{i} \text{ is part of.}$ It actually shows the number of sentences in which a word occurs. While term frequency shows the total count of a word in a text, this shows the number of sentences in which that word occurs. The term frequency can be higher than this since it also counts the word repetitions in a single sentence.

C. Similarity matrix $L$ of $H_{\Gamma}$

Similarity matrix $L$ of the hypergraph $H_{\Gamma}$ is a square matrix, and can be written as

$$L = D_{v} - A \tag{1}$$

D. Diagonal matrix of edge degree

Diagonal matrix of edge degree $D_{e}$ of hypergraph $H_{\Gamma}$ is a square matrix, where the entries are the degree of the hyperedge $\xi_{j}$. The degree of the hyperedge $\xi_{j}$ is equal to the sum of the degrees of all the hypernodes $v_{i}$ in that hyperedge $\xi_{j}$. This is the sum of number of sentences to which each word in a sentence is part of. Suppose a sentence has $n$ words in it. Let $C_{wi}$ be the count of sentences of the text $\Gamma$ in which that word occurs. The edge degree is equal to the sum of $C_{wi}$ of all words in that sentence $S$.

E. Matrix $H$ of weighted hypergraph $H_{w\Gamma}$

The matrix $H$ is the one where the rows represents the hypernodes and the columns represent the hyperedges. An entry $H(v, e, \xi_{j}) = 1$ iff $v_{i}$ is a part of the edge $\xi_{j}$. Inturn it means that a word $w_{i}$ is a part of the sentence $S_{j}$.

F. Weight matrix $W$ of weighted hypergraph $H_{w\Gamma}$

The weight matrix $W$ of a hypergraph $H_{w\Gamma}$ is a diagonal matrix of weights $w$ of hyperedges $\xi_{j}$. The weight $w_{i}$ of a hyperedge $\xi_{j}$ is equal to the sum of the weights of all hypernodes $v_{i}$ in that hyperedge $\xi_{j}$. With respect to the text $\Gamma$ and the hypergraph $H_{\Gamma}$, the weight of a hypernode is the term frequency of the word in the entire document.

G. Matrix $H$ of weighted hypergraph $H_{w\Gamma}$

Matrix $H = V * E \tag{2}$

where $h(v,e) = 1$ if $v$ is an element of $e$, i.e, if a word $w_{i}$ is part of a sentence $S_{j}$. And $h(v,e) = 0$ otherwise, i.e, if a word $w_{i}$ does not occur in a sentence $S_{j}$.

H. Adjacency matrix of weighted hypergraph $H_{w\Gamma}$

Adjacency Matrix

$$A = H^{\top}WHT - D_{v} \tag{3}$$

where $D_{v}$ is the diagonal matrix of node degree of weighted hypergraph which is formed in the same way mentioned in section IV.B.

I. Similarity matrix $L_{w}$ of weighted hypergraph $H_{w\Gamma}$

The similarity matrix $L_{w}$ of weighted hypergraph $H_{w\Gamma}$ can be calculated as

$$L_{w} = D_{v} - A \tag{4}$$

HYPERGRAPH MORPHOLOGY - HYPERGRAPH CONTRACTION

Since spectral partitioning is applied to the fidel vector of the similarity matrix $L$ of the hypergraph $H_{\Gamma}$ of the text $\Gamma$ in method I and weighted hypergraph $H_{w\Gamma}$ in method II, the vector is being divided in to two. Correspondingly two sets of text $\Gamma_{1}$ and $\Gamma_{2}$ are created for the text $\Gamma$. Two hypergraphs $H_{+\Gamma}$ and $H_{-\Gamma}$ are again created with the new set of nodes formed as part of the partitioning. Thus the initial hypergraph is being contracted in each iterative phase.

```
Figure 4: Eigen values

| 0.5576264161 | -0.5865938750 | 0.5753625230 |
| 0.5865938750 | 0.5576264161 | -0.5865938750 |

| 0.5993997290 | -0.3274200000 | 0.6599000000 |
| 0.6599000000 | 0.5993997290 | -0.3274200000 |
```

```
Figure 5: Eigen vector

| -1.1225105640 | 2.0000000000 | 0.0000000000 |
| 2.0000000000 | -1.1225105640 | 0.0000000000 |
```

2130
As mentioned above, this paper has implemented two methods for text clustering, each one again in non weighted and weighted manner. In this paper text clustering refers to clustering text(sentences) in to groups, while in document clustering multiple documents are involved. The eigen vector corresponding to the maximum absolute eigen value is selected as the fidel vector. The same is shown in Fig .4 and Fig .5. The first method in text clustering uses a non weighted hypergraph and spectral partitioning is applied to it based on the sign of fidel vector. The method has many iterations as seen in Fig .6 and Fig .7, until there are no change in signs. Fig .8 shows the positive and negative split made in the selected vector. The figures shown in this paper do not pertain to a single test case, but are outputs of various test cases. The second method uses a weighted hypergraph where the weight of the hyperedge of the graph is sum of the weights of the hypernodes in that edge. In turn the weight of the hypernode will be equal to the number of sentences in which that word occurs. The detailed algorithm can be seen in in the following section.

A. Method I

**Algorithm I**: Clustering text \( \Gamma \) by Spectral partitioning hypergraph \( H_{\Gamma} \).

**Input**: Text to be clustered \( \Gamma \).

**Output**: Clusters.

While (there is no change in sign or there are less than two elements in a group)
1. Create a hypergraph \( H_{\Gamma} \) of the text \( \Gamma \).
2. Create adjacency matrix \( A \) of the hypergraph \( H_{\Gamma} \).
3. Find \( D_{\Gamma} \) of the hypergraph \( H_{\Gamma} \).
4. Find Similarity Matrix \( L \) of \( H_{\Gamma} \).
5. Find the eigen values of \( L \).
6. Select fidel vector corresponding to eigen value \( \lambda_i \).
7. Partition the vector based on the +/- values.
8. Divide the sentences such that all those with negative value fall in one group and those with positive value fall in the second group.

B. Method II

**Algorithm**: Clustering text \( \Gamma \) by using weighted hypergraph \( H_{w\Gamma} \).

**Input**: Text to be clustered \( \Gamma \).

**Output**: Clusters.

while (there is no change in sign or there are less than two elements in a group)
1. Create a weighted hyper graph \( H_{w\Gamma} \) of the text \( \Gamma \).
2. Calculate weight matrix \( W \) of a hypergraph \( H_{w\Gamma} \).
3. Calculate Matrix \( H \) of weighted hypergraph \( H_{w\Gamma} \).
4. Find the matrix \( L_{w} \) of weighted hypergraph \( H_{w\Gamma} \).
5. Find the eigen values \( \lambda_i \) of \( L_{w} \).
6. Select the eigen value with maximum absolute value.
7. Find the fidel vector.
8. Partition the elements in the fidel vector to + and -.
9. Divide the sentences such that all those with negative value fall in one group and those with positive value fall in the second group.

**IMPLEMENTATION**

The implementation of text clustering using hypergraph is done in python. The input document contains Malayalam text. Malayalam is a regional language spoken in Kerala state and Lakshwadeep in India. A Malayalam lemmatizer is developed using a tree based method which reduce the words to its root lemma form. Stop word removal is done before lemmatizing, where some of the verbs and its morphological forms are also included in the stop word list. Some verbs are not considered as they do not contribute much to the meaning of the document. Lemmatizing is essential since the term frequency need to be calculated and moreover the morphological forms of a word may not be available in the dictionary. Only the lemmatized word can be checked against the dictionary. Lemmatizing also
help in identifying the connection between the sentences for the graph creation and the hypergraph creation.

A. Data set
The data set consists of Malayalam articles taken from the Malayalam news sites. These text after preprocessing like punctuation removal, white space removal and stop word removal are subjected to Lemmatization. The text containing lemmatized words are used for creating graph and hypergraph. Both the graphs are subjected to spectral partitioning and results are compared. The clustering applied here help in grouping together news articles related to particular topic. The data set consists of news related to various topics like politics, travel, sports, medical news, film news etc. The work has successfully clustered the articles in to groups which finally resulted in an automatic topic identification. An overview of the dataset is shown in Table I.

<table>
<thead>
<tr>
<th>Categories</th>
<th>Word count</th>
<th>Lemmatized word count</th>
<th>Stop word count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sports, film</td>
<td>700</td>
<td>130</td>
<td>60</td>
</tr>
<tr>
<td>Sports, geography</td>
<td>3000</td>
<td>1000</td>
<td>350</td>
</tr>
<tr>
<td>Politics, geography</td>
<td>4000</td>
<td>1000</td>
<td>160</td>
</tr>
<tr>
<td>Film, politics</td>
<td>5000</td>
<td>1900</td>
<td>320</td>
</tr>
<tr>
<td>Health, kitchen tips</td>
<td>10000</td>
<td>4050</td>
<td>990</td>
</tr>
<tr>
<td>Travel, automobile, gadgets</td>
<td>20000</td>
<td>6400</td>
<td>1400</td>
</tr>
</tbody>
</table>

a. This data set is taken from Malayalam news sites.

B. Performance comparison
Hypergraph modeling of a text can be compared with graph modeling of a text which was followed so far in many languages. A document is modeled as a graph as in Fig.9, by considering the sentences as vertices and the edges exist between two vertices if there is a common word in both sentences. The connectivity of the graph shows which all sentences are being connected and the disconnectivity of the sentences shows isolated sentences. This type of modeling has many disadvantages as the graph which is created does not show how the document is really organized. While the graph method does not convey the idea about which all words make the sentences connected, the hypergraph shown in Fig.1, gives a clear idea about the actual organisation of the document, sentences and the words in it. From Fig .1, the distribution of the words in the document is evident.

The performance of text clustering using graph and hypergraph can be done based on the parameters like speed, accuracy and complexity of the methods. Among the two methods discussed above, the weighted method shows more accuracy than the non weighted method, so also the weighted method is used for comparison with simple graph method. The weight of the edge will be the sum of the weights of the vertices in that edge. In the following sections, the new two methods like hypergraph and weighted hypergraph are compared with contemporary methods like graph and weighted graph.

C. Iterations
The number of iterations in the hypergraph method is more since it identifies the outliers in the data set correctly. Outliers are the text which do not belong to any of the clusters. Outlier detection in the graph method is less. Since the graph modeling using hypergraph is more meaningful with respect to the organisation of the document, outliers can be eliminated more correctly. But since the document modeling using the graph convey less knowledge about the document, the outlier detection and elimination is affected and not according the expectations of the reader. The anonymous sentences eliminated from the hypergraph method always satisfies the reader.

D. Space complexity
The number of edges in the hypergraph representing the text will be equal to the number of sentences considered. If there are 100 sentences to be clustered, whether they are connected or not, the number of edges will be fixed and is equal to 100. But for the graph of a text with 100 sentences, considering connections among all the vertices, the number of edges will be 99 + 98 +...1. Generally we can say that for text with n sentences, the number of edges in a hypergraph is n, while the number of edges in a graph is n-1 + n-2 ....n-(n-1) which can be written as (n-1)n/2. The number of hypernodes in the hypergraph method is equal to the number of unique terms considered for clustering, while the number of vertices in the graph method is equal to the number of sentences considered for clustering. The number of unique terms will be less as we remove the stop words from the text in the preprocessing phase. More over there will be many repeating words in the text. Word repetition can happen in a single sentence and also across sentences. So also the space complexity of the hypergraph is less when compared to the graph.

E. Time Complexity
Time complexity of the both graph and hypergraph methods can be evaluated based on the above two parameters like number of iterations and the number of edges and vertices that need to be constructed in each iteration of the algorithm. Even though there is a slight increase in the number of iterations in the hypergraph method, it is compensated by the reduction in the number of edges and nodes that need to be constructed in every iteration of the algorithm. Even though the number of iterations in the graph method is less, it takes more time because of the increased number of edges and vertices that need to be constructed in each iteration of the algorithm. The time comparison of both graph and hypergraph can be shown as in the Fig .10. Both the plots show a peak value initially in the time of graph creation, because the algorithm creates the original graph of the document in the first step. In each
iteration, due to spectral partitioning, the size of the graph to be created decreases. That is why the execution time for graph creation is minimum in the final iteration. Considering all the iterations, the total time taken for graph creation is 0.0663611889 seconds in the case of graph based method, while the total time taken for hypergraph creation is only 0.0194739911 seconds. This time is for the smallest data set mentioned in Table I.

The results can be tabulated as in Table II. The results are generated for different data sets of varying sizes. The recall is always 1.0 since the false negatives generated by the system is nil, ie, the number of clusters, the system identifies as outliers is nil. The main contributor of the good value of recall, is the efficient outlier detection by the proposed system. Similarly, result analysis is made for the spectral partitioning by the weighted hypergraph. The results obtained are consolidated in Table III. Entropy is a metric that's a measure of the amount of disorder in a vector. Among the various versions of entropy, the one which is selected in the project is Shannon's entropy. Fig .11 has four entropy plots representing graph, weighted graph, hypergraph and weighted hypergraph. Among the four methods weighted hypergraph has the lowest entropy. The graph shows a decrease in the disorder as the number of clusters increase.

**Table II. Result of Spectral Partitioning of Hr**

<table>
<thead>
<tr>
<th>Data set word count</th>
<th>Precision</th>
<th>Recall</th>
<th>Accuracy</th>
<th>Fmeasure</th>
</tr>
</thead>
<tbody>
<tr>
<td>700</td>
<td>0.90</td>
<td>1.0</td>
<td>0.90</td>
<td>0.95</td>
</tr>
<tr>
<td>3000</td>
<td>0.90</td>
<td>1.0</td>
<td>0.92</td>
<td>0.95</td>
</tr>
<tr>
<td>4000</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>5000</td>
<td>0.95</td>
<td>1.0</td>
<td>0.96</td>
<td>0.99</td>
</tr>
<tr>
<td>10000</td>
<td>0.96</td>
<td>1.0</td>
<td>0.97</td>
<td>0.98</td>
</tr>
<tr>
<td>20000</td>
<td>0.98</td>
<td>0.95</td>
<td>0.98</td>
<td>0.96</td>
</tr>
</tbody>
</table>

**Table III. Result of Spectral Partitioning of HWr**

<table>
<thead>
<tr>
<th>Data set word count</th>
<th>Precision</th>
<th>Recall</th>
<th>Accuracy</th>
<th>Fmeasure</th>
</tr>
</thead>
<tbody>
<tr>
<td>700</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>3000</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>4000</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>5000</td>
<td>0.95</td>
<td>1.0</td>
<td>0.96</td>
<td>0.98</td>
</tr>
<tr>
<td>10000</td>
<td>0.98</td>
<td>1.0</td>
<td>0.98</td>
<td>0.99</td>
</tr>
<tr>
<td>20000</td>
<td>0.99</td>
<td>1.0</td>
<td>0.99</td>
<td>0.99</td>
</tr>
</tbody>
</table>

The hypergraph method give more accurate results in the case of cluster formation and in the case of outlier detection. The clusters which end up with single sentences are considered outliers and others are actual clusters which speak about a particular topic. The accuracy is being tested manually with native Malayalam news readers. The clusters returned by the algorithm is compared against the clusters returned manually by the native people. The number of clusters returned by the weighted hypergraph method shows an accuracy of the 98% and the clusters returned by the contemporary graph method shows an accuracy near to 80%. The precision, recall, fmeasure of the clustering is calculated based on the true positives, true negatives, false positives and false negatives. True positive is defined as the number of clusters correctly assigned by the method. True negatives are defined as the number of outliers correctly identified by the system, False positives are the number of outliers that the system marked as clusters and the False negatives are the number of clusters, the system marked as outliers. Once these measures are obtained, the following are calculated

\[
\text{Precision} = \frac{t_p}{t_p + f_p}
\]

(5)

\[
\text{Recall} = \frac{t_p}{t_p + f_n}
\]

(6)

\[
\text{Accuracy} = \frac{t_p + t_n}{\text{Total}}
\]

(7)

\[
\text{Fmeasure} = 2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall})
\]

(8)

**F. Accuracy and Entropy**

Figure 10: Comparison of graph creation time

The results can be tabulated as in Table II. The results are generated for different data sets of varying sizes. The recall is always 1.0 since the false negatives generated by the system is nil, ie, the number of clusters, the system identifies as outliers is nil. The main contributor of the good value of recall, is the efficient outlier detection by the proposed system. Similarly, result analysis is made for the spectral partitioning by the weighted hypergraph. The results obtained are consolidated in Table III. Entropy is a metric that's a measure of the amount of disorder in a vector. Among the various versions of entropy, the one which is selected in the project is Shannon's entropy. Fig .11 has four entropy plots representing graph, weighted graph, hypergraph and weighted hypergraph. Among the four methods weighted hypergraph has the lowest entropy. The graph shows a decrease in the disorder as the number of clusters increase.
MULTIDOCUMENT CLUSTERING

The same two methods can be used for multi document clustering. The hyper graph is constructed by modeling documents as hyperedges and unique words as hypernodes. The above mentioned two methods of spectral partitioning can be applied where the hypernodes are partitioned based on the sign of the elements in the fidel vector. All hypernodes(words) which are associated with negative values form one cluster and those with positive values fall in the second cluster. All documents(edges) with those nodes with negative values fall in one cluster and those with positive values fall in the second one. This is iteratively done until there is no change in sign or the cluster has only one document in it.

CONCLUSION

The paper presents a novel method of modeling text using hypergraphs and weighted hypergraphs. The sentences in the text forms the edge set and the words in the text forms the node set. Once the hypergraph is constructed various mathematical operations like finding the adjacency matrix, diagonal matrix of node degree, diagonal matrix of edge degree, similarity matrix matrix $H$, weight matrix are calculated. On applying spectral partitioning to the similarity matrix, the edges could be divided into partitions where by it results in a hypergraph morphology named hypergraph contraction. Hypergraph contraction leads to the formation of text clusters. Both hypergraph method and weighted hypergraph methods are being compared with existing graph and weighted graph method. Hypergraph methods saves time in graph creation and clustering by 29% against the contemporary graph methods. Hypergraph method also saves space since only less space required in each iteration. More over the accuracy of the text clustering is more in the case of hypergraphs. While hypergraph method shows an accuracy of 95.5%, weighted hypergraph partitioning shows an accuracy of 98%.

REFERENCES


