Content Based Partitioning using Spectral Approach in Social-Networks

S Pradeepa, Vindhya Guru Rao, Diksha Shree, Sindhuja S

1Assistant professor-II, School of Computing, SASTRA Deemed to be University, Thanjavur, India
2,3,4Student, School of Computing, SASTRA Deemed to be University, Thanjavur, India

pradeepa@it.sastra.edu, vindhyarao97@gmail.com, shreediksha99@gmail.com, sindhuja361997@gmail.com

Abstract—Partitioning in a Social-network has become an important task for data dissemination. With segregation of users we can identify matter of interests and establish similarity amongst users. In this study, we propose an effective way of partitioning for identifying user groups in a social networking site. We propose a framework to find the partitions based on the similarities in the tweets. Firstly, similarity is established between the contents. Then, we propose a strategy to find the partitions based on the similarity and hence further group them. In this way, we investigate the social community partition problem which is one of the most effective ways to communicate and share common ideas.

Keywords—partitioning, social-networking, tweets, eigenvalues, eigenvectors, spectral clustering

I. INTRODUCTION

The technology has expanded at a large scale these days especially in the field of web and internet related high tech. This growth entertains the user in various areas like e-commerce or any networking sites. Hence, there is a need of some recommendation system to find what user wants and their preferences for any item. In particular, user needs such a system that reduces their time to search and update new products. Those products will be completely based on their precedence. Various approaches have been adopted to improve the performance of recommendation system where content based recommendation system is one such approach which intends to help user choose their item and its related items. Graph based approaches have prevailed – Content based, graph based, collaborative filtering. A content based recommendation system recommends items based on a comparison between the contents of the item. The content of each item is represented as some term value, a feature, typically the count of each item in a document. With the help of this count, this system builds the overall analysis for each item in a document and calculates its similarity with other items, which gives the occurrence of that item in the user profile. In the present study, we take the “twitter samples” as a data set consisting of as many tweets users post. Then the tweets which are similar to any other tweets in a document are calculated. Based on those values, the network is formed. That is, users, are grouped together to form large networks. A connected component of this network is a subgraph of nodes (users), who tweet on similar contents and hence a topic of interest when recommending, can be recommended to all the users of the found community. The community is found through partitioning based on Spectral clustering or spectral graph partitioning methods, explored in detail in the further sections. This method is an example of the Hybrid recommendation system, that uses contents and with graph-based techniques, allows identification of communities.

II. RELATED WORKS

A very established usage of spectral clustering has been made in image segmentation, including the optimization of the cut to determine the best possible decomposition of any image. Application of similar eigenvector selection in spectral clustering is also seen in pattern recognition techniques. To the best of our efforts, we found that this method has not yet been applied or recognized extensively in social...
networking where the networks play an important role in dissemination of data.

Approaches have been made in recommendation systems defining multi-attribute networks [2] using degree centrality and closeness centrality [3]. This paper of Newman [3], talks of centrality measure that can be used in determining importance of a node in the network [4], stating more the connections, more the power the node holds. Also, eigenvector centrality is stated saying that not all connections are equal. The number and the quality of connections both play an “important” role. The level of influences varies from center to center.

Establishing just semantic similarity [5] cannot be solely used where there is a network of users and data dissemination happening. Here is where the usage of partitioning or clustering algorithms has become an important step for further analyses.

Other well-known methods for clustering are k-means clustering and agglomerative clustering. Various kernel functions operate on the affinity matrix of datasets, to result in clusters. These clusters are formed based on the closeness to each other based on the Euclidean distances [20]. This method has had its disadvantages with processing of data that are non-linearly separable [16] and also in formation of varied density clusters. The agglomerative clustering technique follows a bottom-up approach that merges the pair of clusters, based on Euclidean linkages, recursively, increasing the linkage distance minimally. Spectral approach is found to give better results as we work on Twitter content, and compare this method, kernel k-means approach and agglomerative clustering for Content-Based Partitioning in this paper.

III. PROPOSED METHOD

We propose a Content Based Filtering approach that analyses the relationships between items on a broad aspect based on their various features. These various features carry “importance” that we evaluate using suitable similarity algorithm. This importance is graded and threshold to establish similarity amongst items. Broadly, the overall algorithm is as follows: A. Collect items and their features, B. Establish similarity amongst items using importance value of their features, C. Provide a threshold on the similarity to form the network-a graph, D. Item partitioning it to identify groups.

A. Collection of items and their features

An item is any considerable entity, expected to be analyzed. The features are the characteristics that describe this item. For instance, consider a document to be our item. We expect to ultimately establish similarity between this document and other documents. This document will have terms - with their relevance in the document. These terms are of importance and form the “feature” of the item. Our algorithm collects as many features as possible for every item, as the relevance between pair of documents and also amongst documents is to play an important role in clustering and for further recommendations.

An item-feature relationship is represented as a matrix of items versus the total of all features (terms of relevance).

<table>
<thead>
<tr>
<th>Document/Term</th>
<th>Term 1</th>
<th>Term 2</th>
<th>Term 3</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document 1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>…</td>
</tr>
<tr>
<td>Document 2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>…</td>
</tr>
</tbody>
</table>

B. Establishing item similarity

One of the most popular term-weighting schemes is TF-IDF.

1) Term frequency (TF)

A central question in Natural Language Processing is how to quantify what a document is about. This can be done by looking at the words in the document. One measure of how important a word may be is its term frequency (TF), how frequently a word occurs in a document. When a word that properly makes sense occurs in a document many times, it obviously means that it is important to the document. We calculated the term frequency with (1). The TF value of a term ‘t’ in a document ‘d’ is given by the frequency ‘f’ of that term in the document divided by the number of words in that document.

\[
TF_{(t,d)} = \frac{f(t,d)}{\text{number of words in } d}
\] (1)

The term frequency cannot fully decide the important words of a document because words like ‘the’, ‘is’, ‘of’, ‘in’ etc. will have high term frequency. We might take the approach of adding words like these to a list of stop words and removing them before analysis, but it is possible that some of these words might be more important in some documents than others. A list of ’stopwords’ is not a sophisticated approach to adjusting term frequency for commonly used words.

2) Inverse Document Frequency (IDF)

Another approach is to look at a term’s inverse document frequency (IDF), which decreases the weight for commonly used words and increases the weight for words that are not used very much in a collection of documents. That is, whether the term is common or rare across all documents. We calculated the inverse document frequency with (2).
\[ IDF(t) = \ln \left( \frac{n_{\text{documents containing } t}}{n_{\text{documents}}} \right) \]  

(2)

3) Term Frequency-Inverse Document Frequency (TF-IDF)

In information retrieval, term frequency-inverse document frequency i.e., TF-IDF, is a numerical statistic that is intended to reflect how important a word is to a document. It is this TF-IDF value which gives the clear picture about the important words.

\[ TF = IDF_{\text{word}} \times IDF_{\text{word}} \]  

(3)

The TF-IDF will give the set of important words of any document in a collection or corpus.

4) Jaccard Coefficient

The Jaccard coefficient is the measure used for comparing the similarity between documents. Now, the similarity or utility matrix is calculated to find the set of words that are similar in the document based on TF-IDF value. The data in utility matrix gives the degree of preference of items of that user for that item.

Utility matrix is formed by finding the Jaccard coefficient. Then Jaccard coefficient is calculated as in (4) or (5)

\[ J(A,B) = \frac{|A \cap B|}{|A \cup B|} \]  

(4)

\[ J(A,B) = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \]  

(5)

\( J(A,B) \) refers to similarity between any two documents A and B based on the number of same words present in both the documents (after elimination of unimportant words using TF-IDF) upon all the words in both of them.

When calculated for every pair of document, I can result as the item-to-item similarity matrix, with the values ranging from 0 to 1.

C. Construction of the network

Partitioning has become important in the field where networks of entities exist and is a step ahead of establishing just semantic similarity[5] and user modelling[6]. This part of the algorithm involves two basic steps, broadly written: 1) Providing the threshold[13][7] to establish the similarity, 2) Generating the item-network based on the most relevant items connected together (the threshold).

The similarity matrix so obtained gives similarity between every pair of items. But when handling data in enormous amounts, multiple relationships exist between any pair of data, or rather, all data is collaborated. Therefore for a better analysis, a network (a graph) of the items is built, where the relationships are expressed as network edges. This gives the overall picture of the relationship between the items. Broad conclusions can already be inferred from this. A dense graph represents strongly connected documents, indicating greater similarity amongst the contents. A sparse graph represents loosely connected data. This clearly, provides a structure to the items, letting us perform the network analysis in a more efficient manner.

1) The setting of threshold

Where the range of values is between 0 and 1, the threshold could be set as 0.049, denoting that he expects more similarity than when setting it as 0.03, even when the chances of getting more number of similar items pertains in the latter case.

2) Creating the graph

Once this threshold is decided, item to item relationship is established in the graph as node to node with the weight of the edge representing the similarity value. This is the weighted graph representation of the acquired data. A network is thus formed of nodes, connected to their neighbour nodes that are more similar than the others.

Analyzing complex networks needs a measure. This measure is the modularity[8][9] metric introduced by Newman. This helps us quantify group of nodes in a network, and hence form the community. They define relevance and closeness of a certain feature in the community.

D. Partitioning

This is based on graph-based network analysis. This analysis involves the following aspects: matrix theory, minimization of the edges (graph theory), minimization of the cut (the partitioning strategy). The cut mentioned here is nothing but the formation of partitions in the network. This is done so as to identify the user groups looking for similar contents. When such user groups are identified, it is easier for the recommendation systems to provide relevant suggestions to the communities.

A partitioned structure from the graph uses the existence of edges between any two nodes of the network. It is thus meaningful to assume that, if the number of edges within groups is significant then the constituting nodes are more closely related, into a community.

From the graph theory, we know that adjacency matrix (A), degree matrix (D), are the ones which describe the graph. The Laplacian matrix (L) obtained by: L = D - A, is the one used to perform the analysis on the network.

The use of spectral methods to compute edge separators in graphs was first considered by Donath and Hoffman are the first to suggest using the eigenvectors of adjacency matrices of graphs to find partitions. Since then spectral methods for computing various graph parameters have been considered by several others [14]-[17]. Using spectral methods, edge separators, or what we call cuts, in graphs could be computed.

The modularity of the graph is the descriptor of the number of connected components in the whole network. It is calculated as Q = X^TLX, where X is the column vector in the graph Laplacian. In many approaches of Spectral Clustering, instead of the laplacian, the adjacency matrix has been used. Further procedure according to the standard spectral partitioning methods is stated ahead.

With further developments in the algorithm, the usage of normalized adjacency matrix (A') has shown better results [1][14].
It is calculated using (6):
\[ A' = L^{-1/2}AL^{-1/2} \]  

(6)

Let the eigenvalues of A’ be ordered as suitable, inferred from \[\lambda_0 < \lambda_1 < \lambda_2 ... < \lambda_n \]

An eigenvector corresponding to \(\lambda_0 \) is vector of all ones.

The multiplicity of \(\lambda_0 \) is equal to the number of connected components of the graph. For a connected graph, the second smallest eigenvalue \(\lambda_1 \) is greater than zero.

The following algorithm, as suitable, inferred from [10]-[12] is written to bisect the graph, by separating the vertices into 2 groups, and further recursively partition to receive required number of graph components, or communities.

**Algorithm:** Content based partitioning

**Input:** Similarity matrix \(S \in \mathbb{R}^{n \times n} \), number \(k \) of clusters to construct.

1. Construct a similarity graph.
2. Set a suitable threshold, above which two nodes are joined by edges, thus form a graph
   Let \(W \) be its weighted adjacency matrix.
3. Compute the normalized Adjacency \(A’\) of \(A\) using (6)
4. Compute the first \(k\) eigenvectors \(u_1\ldots u_k\) of the generalized Eigen problem \(A’X = \lambda X\).
5. Let \(U \in \mathbb{R}^{k \times n}\) be the matrix containing the vectors \(u_1\ldots u_k\), as columns.
6. For \(i = 1\ldots n\), let \(y_i \in \mathbb{R}^k\) be the vector corresponding to the \(i^{th}\) row of \(U\).
7. Cluster the points \((y_i)_{i=1...n}\) in \(\mathbb{R}^k\) with the Spectral k-means algorithm into clusters \(C_1 \ldots C_k\).

**Output:** Clusters \(A_1\ldots A_k\) with \(A_i = \{j | y_j \in C_i\}\).

Determining the best possible, or the minimized cut is an NP-complete problem [1]. Approaches were made to find the best possible cut. The algorithms have been codified and are available as coded libraries in established software like MATLAB and Python (Sci-kit learn)[1][11].

**IV. EXPERIMENTAL ANALYSIS**

We implemented our approach with the "twitter_samples" corpus available in the nltk package. We have coded using Python. Here, we consider tweets to be documents. At first, we find the bag of words of each tweet and hence the bag of words of the whole corpus if found. Bag of words of a corpus is a multiset of its words. Then, we calculate the TF, IDF, TF-IDF and Jaccard coefficient values.

**A. Term frequency (TF) calculation:**

From the "twitter_samples" corpus, the frequency of every word in the document was found. Then, the term frequency was calculated by dividing the frequency of the word by the count of the bag of words of the document. Thus, each document returns an array of term frequency values.

**B. Inverse document frequency (IDF) calculation:**

Now that the frequency of each term has been calculated, as mentioned earlier, it might also lead to adding function words to the list of the features. To eliminate that, we made use of this factor. The IDF value of a word was calculated by dividing the total number of the documents by the number of the documents containing the word and taking the logarithm value of this result.

**C. Term frequency-Inverse document frequency (TF-IDF) calculation:**

The TF-IDF value is the product of both the TF and IDF values. A suitable threshold was set, to eliminate the unimportant and unnecessarily redundant words, leaving essential words for further similarity comparison amongst tweets.

**D. Jaccard similarity:**

One of the most popular methods to measure similarity is the Jaccard similarity or Jaccard coefficient. The Jaccard coefficient value is calculated between two documents (tweets) at a time. The steps are, for experimental calculations:

1. Count the number of members which are shared between both sets.
2. Count the total number of members in both sets (shared and unshared).
3. Divide the number of shared members by the total number of members.

The members are the words of the tweets here. Thus, the Jaccard coefficient values are found and a similarity matrix is created using the values. Figure 2 shows the result of TF-IDF and Jaccard coefficient calculations on 6 sample data from the corpus.

Clearly, this is a symmetric matrix, each value defining the similarity between every pair of items (tweets).

This is also called the affinity matrix and according to [1], [11] application in the sklearn.cluster library, the algorithm can also be used to find the normalized cut.

The graph plotted of the sample data set based on 100 tweets of the nltk corpus twitter_samples is shown in Figure 3.
The Spectral Clustering algorithm using the eigenvalues was applied on the adjacency matrix (also symmetric) built based on the similarity matrix. The nodes were plotted, results shown in Figure 4.

For the same dataset Kernel K-means was applied to obtain the result as in Figure 5. Figure 4 and Figure 5 are results when the number of cluster argument of the library function ‘n_clusters’ was set as 2.
<table>
<thead>
<tr>
<th>Feature/Method</th>
<th>Kernel K-means</th>
<th>Agglomerative Method</th>
<th>Spectral Clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stability</td>
<td>Less stable</td>
<td>Stability is good</td>
<td>Better stability</td>
</tr>
<tr>
<td>Cut</td>
<td>Not optimal</td>
<td>Minimized</td>
<td>Minimized</td>
</tr>
<tr>
<td>Distribution</td>
<td>Very uneven</td>
<td>Uneven</td>
<td>Uniform</td>
</tr>
<tr>
<td>Performance</td>
<td>Less ideal</td>
<td>Does not guarantee</td>
<td>Can handle</td>
</tr>
<tr>
<td></td>
<td>good performance</td>
<td></td>
<td>large amount of data</td>
</tr>
</tbody>
</table>

Clear bisection into group is visible with Spectral Clustering using eigenvalues, as a suitable normalized cut was applied. But in the case of Kernel K-means, the two clusters formed are uncertainly cut, and minimization of cut as an objective, is unsuccessful. Though the cut is minimized with agglomerative clustering, the bisection was found to be uneven (TABLE 2). To establish better, the efficiency of Spectral Clustering using eigenvalues and eigenvectors over Kernel K-means Clustering, results of both these methods for number of clusters set as 4 are also given in figures 8 and 9. Figure 10 and Figure 11 show the distribution of the nodes into 4 clusters. Moreover, with the Spectral Approach when used for partitioning based on content, gives a better bisection as compared to the result of agglomerative clustering for the same content (Figures 12 and 13).

V. CONCLUSION

This method of partitioning is very useful to identify different categories of users in different fields and thus, look up their preferences which simplify their task. Generally, users find it difficult to get the updates of their choice from some millions of statuses. This content based system will help the user to address the issue of information overloading. Recommendation systems face many difficulties to implement partitioning logic in real-time usage. We gave the easier and effective way for a user to proceed with his precedence and let him choose his own interest. We divided the users into different groups based on the similarity in their tweets. Graph which was formed with the help of similarity matrix is then partitioned to obtain the ultimate grouping. The threshold value is set up to start creating the graph. The spectral clustering algorithm was applied to find the eigenvalues which further gives the partition. Since this is a graph based approach, it is less tedious and yet standard in partitioning into groups. The whole method of partitioning is much simpler and effective than the Kernel K-means clustering method and also agglomerative clustering, and substantial proofs were provided to establish this fact.

FUTURE WORK

Partitioning and clustering have a wide range of applications, and a content based approach for this can be expanded to the medical field where possible correlated diseases or symptoms can be identified. If implemented well in future, this could help ease the process of medical diagnosis.
REFERENCES


