IJDCST @November Issue- V-1, I-7, SW-65
ISSN-2320-7884 (Online)
ISSN-2321-0257 (Print)

Different View Points Measuring the Similarities in Web Based Documents

M.Valia Kumar[1], M.Veerabhadra Rao[2]
Dept of Computer Science and Engineering[1,2]
Prasiddha College of Engineering & Technology

Abstract- Measuring similarity or distance between two entities is a key step for several data mining and knowledge discovery tasks. The notion of similarity for continuous data is relatively well-understood, but for categorical data, the similarity computation is not straightforward. Several data-driven similarity measures have been proposed, The existing algorithms for text mining make use of a single viewpoint for measuring similarity between objects. Their drawback is that the clusters can't exhibit the complete set of relationships among objects. To overcome this drawback, we propose a new similarity measure known as Hierarchical multi-viewpoint based similarity measure to ensure the clusters show all relationships among objects. We also proposed two clustering methods. The empirical study revealed that the hypothesis "multi-viewpoint similarity can bring about more informative relationships among objects and thus more meaningful clusters are formed" is proved to be correct and it can be used in the real time applications where text documents are to be searched or processed frequently**.**

Keywords: cluster, similarity measure, text documents

I. INTRODUCTION

Measuring similarity or distance between two data points is a core requirement for several data mining and knowledge discovery tasks that involve distance computation. Examples include clustering (kmeans), distance-based outlier detection, classification, and several other data mining tasks. These algorithms typically treat the similarity computation as an orthogonal step and can make use of any measure.

For continuous data sets, the Minkowski Distance is a general method used to compute distance between two multivariate points. In particular, the Minkowski Distance of order 1 (Manhattan) and order 2 (Euclidean) are the two most widely used distance measures for continuous data. The key observation about the above measures is that they are independent of the underlying data set to which the two points belong. Several datadriven measures, such as Mahalanobis Distance, have also been explored for continuous data.

Clustering in general is an important and useful technique that automatically organizes a collection with a substantial number of data objects into a much smaller number of coherent groups.

Text document clustering groups similar documents that to form a coherent cluster, while documents that are different have separated apart into different clusters. However, the definition of a pair of documents being similar or different is not always clear and normally varies with the actual problem setting. For example, when clustering research papers, two documents are regarded as similar if they share similar thematic topics. When clustering is employed on web sites, we are usually more interested in clustering the component pages according to the type of information that is presented in the page. For instance, when dealing with universities' web sites, we may want to separate professors' home pages from students' home pages, and pages for courses from pages for research projects. This kind of clustering can benefit further analysis and utilize of the dataset such as information retrieval and information extraction, by grouping similar types of information sources together.

Accurate clustering requires a precise definition of the closeness between a pair of objects, in terms of either the pairwised similarity or distance. A variety of similarity or distance measures have been proposed and widely applied, such as cosine similarity and the Jaccard correlation coefficient. Meanwhile, similarity is often conceived in terms of dissimilarity or distance as well [5]. Measures such as Euclidean distance and relative entropy have been applied in clustering to calculate the pair-wise distances.

A common approach to the clustering problem is to treat it as an optimization process. An

optimal partition is found by optimizing a particular function of similarity (or distance) among data. Basically, there is an implicit assumption that the true intrinsic structure of data could be correctly described by the similarity formula defined and embedded in the clustering criterion function. Hence, effectiveness of clustering algorithms under this approach depends on the appropriateness of the similarity measure to the data at hand. For instance, the original *k*-means has sum-of-squared-error objective function that uses Euclidean distance. In a very sparse and high dimensional domain like text documents, spherical *k*means, which uses cosine similarity instead of Euclidean distance as the measure, is deemed to be more suitable[3], [4].

The work in this paper is motivated by investigations from the above and similar research findings. It appears to us that the nature of similarity measure plays a very important role in the success or failure of a clustering method. Our first objective is to derive a novel method for measuring similarity between data objects in sparse and high dimensional domain, particularly text documents. From the proposed similarity measure, we then formulate new clustering criterion functions and introduce their respective clustering algorithms, which are fast and scalable like *k*-means, but are also capable of providing high-quality and consistent performance.

II.SIMILARITIES MEASURE

Before clustering, a similarity/distance measure must be determined. The measure reflects the degree of closeness or separation of the target objects and should correspond to the characteristics that are believed to distinguish the clusters embedded in the data. In many cases, these characteristics are dependent on the data or the problem context at hand, and there is no measure that is universally best for all kinds of clustering problems.

Measuring similarities between objects differenty ways . Its given below

Metric

Not every distance measure is a metric. To qualify as a metric, a measure d must satisfy the following four conditions.

Let x and y be any two objects in a set and d(x, y) be the distance between x and y.

1. The distance between any two points must be nonnegative, that is, d(x, y)>= 0.
2. The distance between two objects must be zero if and only if the two objects are identical, that is, d(x, y) = 0 if and only if x = y.
3. Distance must be symmetric, that is, distance from x to y is the same as the distance from y to x, ie. d(x, y) = d(y, x).
4. The measure must satisfy the triangle inequality, which is d(x, z) <= d(x, y) + d(y, z).

Euclidean Distance

Euclidean distance is a standard metric for geometrical problems. It is the ordinary distance between two points and can be easily measured with a ruler in two- or three-dimensional space. Euclidean distance is widely used in clustering problems, including clustering text. It satisfies all the above four conditions and therefore is a true metric. It is also the default distance measure used with the K-means algorithm.

Measuring distance between text documents, given two documents da and db represented by their term vectors ta and tb respectively, the Euclidean distance of the two documents is defined as

$$D_E(\overrightarrow{t_a}, \overrightarrow{t_b}) = \left(\sum_{t=1}^{m} |w_{t,a} - w_{t,b}|^2\right)^{1/2},$$

Cosine Similarity

When documents are represented as term vectors, the similarity of two documents corresponds to the correlation between the vectors. This is quantified as the cosine of the angle between vectors, that is, the so-called cosine similarity. Cosine similarity is one of the most popular similarity measure applied to text documents, such as in numerous information retrieval applications [2] and clustering too [9].

Given two documents ta and tb , their cosine similarity is

$$SIM_C(\overrightarrow{t_a}, \overrightarrow{t_b}) = \frac{\overrightarrow{t_a} \cdot \overrightarrow{t_b}}{|\overrightarrow{t_a}| \times |\overrightarrow{t_b}|},$$

where ta and tb are m-dimensional vectors over the term set T = {t1, . . . , tm}. Each dimension represents a term with its weight in the document, which is non-negative. As a result, the cosine similarity is non-negative and bounded between [0,1].

Jaccard Coefficient

The Jaccard coefficient, which is sometimes referred to as the Tanimoto coefficient, measures similarity as the intersection divided by the union of the objects. For text document, the Jaccard coefficient compares the sum weight of shared terms to the sum weight of

terms that are present in either of the two document but are not the shared terms. The formal definition is:

$$SIM_J(\vec{t_a}, \vec{t_b}) = \frac{\vec{t_a} \cdot \vec{t_b}}{|\vec{t_a}|^2 + |\vec{t_b}|^2 - \vec{t_a} \cdot \vec{t_b}}.$$

The Jaccard coefficient is a similarity measure and ranges between 0 and 1. It is 1 when the ta = tb and 0 when ta and tb are disjoint, where 1 means the two objects are the same and 0 means they are completely different. The corresponding distance measure is DJ = 1 − SIMJ and we will use DJ instead in subsequent experiments.

Pearson Correlation Coefficient

Pearson's correlation coefficient is another measure of the extent to which two vectors are related. There are different forms of the Pearson correlation coefficient formula. Given the term set T = {t1, . . . , tm}, a commonly used form is

$$SIM_P(\vec{t_a}, \vec{t_b}) = \frac{m \sum_{t=1}^{m} w_{t,a} \times w_{t,b} - TF_a \times TF_b}{\sqrt{[m \sum_{t=1}^{m} w_{t,a}^2 - TF_a^2][m \sum_{t=1}^{m} w_{t,b}^2 - TF_b^2]}}$$

where $TF_a = \sum_{t=1}^{m} w_{t,a}$ and $TF_b = \sum_{t=1}^{m} w_{t,b}$.

This is also a similarity measure. However, unlike the other measures, it ranges from +1 to −1 and it is 1 when ta = tb . In subsequent experiments we use the corresponding distance measure, which is DP = 1−SIMP when SIM>= 0 and DP = |SIMP | when SIMP < 0.

III. EXISTING SYSTEM

The existing system similarities measure in only one view such as euclidean distance or cosine similarities or jaccard coefficient so it measures similarities between objects only 50 to 60 %.

IV. PROPOSED SYSTEM

Agglomerative hierarchical clustering is a bottom-up clustering method where clusters have sub-clusters, which in turn have sub-clusters, etc. The classic example of this is species taxonomy. Gene expression data might also exhibit this hierarchical quality (e.g. neurotransmitter gene families). Agglomerative hierarchical clustering starts with every single object (gene or sample) in a single cluster. Then, in each successive iteration, it agglomerates (merges) the closest pair of clusters by satisfying some similarity criteria, until all of the data is in one cluster.

Hierarchical Clustering Algorithm – Multi View Point Similarities

- Assign each object to a separate cluster.
- Evaluate all pair-wise similarities using Euclidian distance or cosine similarity or jacard coefficient .
- Construct a distance matrix using the similar object values.
- Look for the pair of clusters with the most similar objects.
- Remove the pair from the matrix and merge them.
- Evaluate all similarities measure from this new cluster to all other clusters, and update the matrix.
- Repeat until the similarity matrix is reduced to a single element.

IV. EXPERIMENTAL RESULTS

To demonstrate how well MVSCs can perform, we compare them with six other clustering methods on the twenty datasets in Table 1. In summary, the eight clustering algorithms are:

- MVSC-*IR*: MVSC using criterion function *IR*
- MVSC-*IV* : MVSC using criterion function *IV*
- k-means: standard k-means with Euclidean distance
- Sp*k*means: spherical *k*-means with CS
- graphCS: CLUTO's graph method with CS
- graphEJ: CLUTO's graph with extended Jaccard
- MMC: Spectral Min-Max Cut algorithm [10]
- HC:Hierarchical Clustering

Many clustering algorithms require parameter to be chosen to determine the granularity of the result. Partitioning methods such as the k-means and k-medoids algorithms require that the number of clusters, k, be specified. Density-based methods use input parameters that relate directly to cluster size rather than the number of clusters. Hierarchical methods avoid the need to specify either type of parameter and instead produce results in the form of tree structures that include all levels of granularity. When generalizing partitioning-based methods to hierarchical ones, the biggest challenge is the performance.

Hierarchical clustering as a search for equilibrium cluster centers requires us to have a fast method of finding data points based on their feature attribute values. Density-based algorithms such as DENCLUE achieve this goal by saving data in a

special data structure that allows referring to neighbors. We use a data structure, namely a Peano Count Tree (or P-tree) [11, 12, 13, 14, 15] that allows fast calculation of counts of data points based on their attribute values.

P-TREE

Many types of data show continuity in dimensions that are not themselves used as data mining attributes. Spatial data that is mined independently of location will consist of large areas of similar attribute values. Data streams and many types of multimedia data, such as videos, show a similar continuity in their temporal dimension. Peano Count Trees are constructed from the sequences of individual bits, i.e., 8 P-trees are constructed for byte-valued data. Compression is achieved by eliminating nodes that consist entirely of 0- or 1-values. Two and more dimensional data is traversed in Peano order, i.e., recursive raster order. This ensures that continuity in all dimensions benefits compression equally. Counts are maintained for every quadrant. The P-tree for an 8-row-8-column bit-band is shown in Figure 1

Hierarchical clustering algorithm that is based on some of the same premises as well-known partition- and density-based techniques. The time-complexity of k-medoids related algorithms is avoided in a systematic way and the influence of outliers is reduced. The hierarchical organization of data represents information at any desired level of granularity and relieves the user from the necessity of selecting parameters prior to clustering. Different levels in the hierarchy are efficiently calculated by using lower level solutions as starting points for the

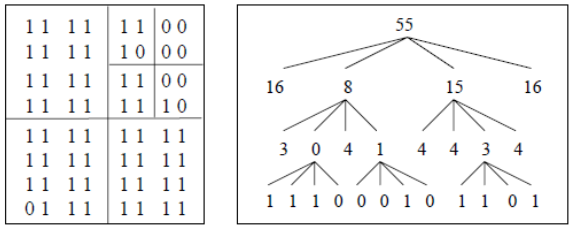

Fig 1: 8x8 image and P-Tree.

computation of higher level cluster centers. We use the P-tree data structure for efficient storage and access of data. Comparison with kmeans shows that we can achieve the benefits of improved outlier handling without sacrificing performance.

We tested the speed and effectiveness of Hierarchical clustering algorithm by comparing with the result of using k means clustering. The data was generated with no assumptions on continuity in the structural dimension (e.g., location for spatial data, time for multimedia data). Such continuity would significantly benefit from the use of P-tree methods. The speed demonstrated in this section can therefore be seen as an upper bound to the time complexity. Speed comparison was done on data with 2 attributes for a range of data set sizes.[15]

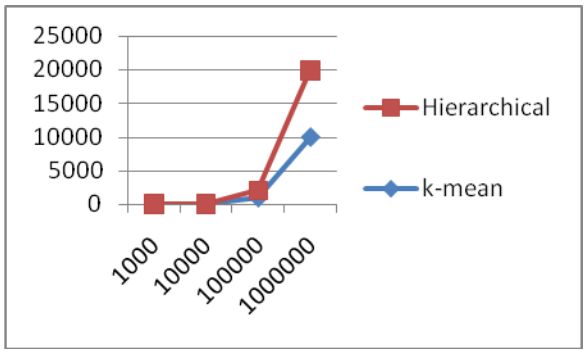

Fig 2: Speed Comparision Heirarchical and K-means approach

The Table 1 compares the cluster centers of k-means for k = 5 with those found by Hierarchical algorithm. K-means results were significantly more influenced by the noise between the identifiable clusters than the results of our algorithm.

Table 1: Comparision of cluster center for the data set of Fig 2:

k-means (k=5)	X	11	4	35	4	23
	y	11	12	6	22	23
Hierarchical	X	9	27	24	4	18
	y	11	22	6	21	25

Clustering solution is evaluated by comparing the documents' assigned labels with their true labels provided by the corpus. Three types of external evaluation metric are used to assess clustering performance. They are the *FScore*, Normalized Mutual Information (*NMI*) and *Accuracy*. *FScore* is an equally weighted combination of the "precision" (*P*) and "recall" (*R*) values used in information retrieval. Given a clustering solution, *FScore* is determined as:

$$FScore = \sum_{i=1}^{k} \frac{n_i}{n} \max_j (F_{i,j})$$

$$\text{where } F_{i,j} = \frac{2 \times P_{i,j} \times R_{i,j}}{P_{i,j} + R_{i,j}}; P_{i,j} = \frac{n_{i,j}}{n_j}, R_{i,j} = \frac{n_{i,j}}{n_i}$$

Where *ni* denotes the number of documents in class *i*, *nj* the number of documents assigned to cluster *j*, and *ni,j* the number of documents shared by class *i* and cluster *j*. From another aspect, *NMI* measures the information the true class partition and the cluster assignment share. It measures how much

knowing about the clusters helps us know about the classes:

$$NMI = \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} n_{i,j}\log\left(\frac{n\cdot n_{i,j}}{n_i n_j}\right)}{\sqrt{\left(\sum_{i=1}^{k} n_i \log\frac{n_i}{n}\right)\left(\sum_{j=1}^{k} n_j \log\frac{n_j}{n}\right)}}$$

Finally, *Accuracy* measures the fraction of documents that are correctly labels, assuming a one-to-one correspondence between true classes and assigned clusters. Let q denote any possible permutation of index set $\{1, \ldots, k\}$,*Accuracy* is calculated by:

$$Accuracy = \frac{1}{n}\max_{q}\sum_{i=1}^{k} n_{i,q(i)}$$

The best mapping q to determine *Accuracy* could be found by the Hungarian algorithm2. For all three metrics, their range is from 0 to 1, and a greater value indicates a better clustering solution.

It can be observed that MVSC-*IR* and MVSC-*IV* perform consistently well. In Fig. 1 19 out of 20 datasets, except *reviews*, either both or one of MVSC approaches are in the top two algorithms. The next consistent performer is Hierarchical Clustering. The other algorithms might work well on certain dataset. For example, graphEJ yields outstanding result on *classic*; graphCS and MMC are good on *reviews*.

The observation, which is also the main objective of this empirical study, is that by applying MVSC to refine the output of spherical *k*-means, clustering are improved significantly. Both rMVSC-I_R and rMVSC-I_V lead to higher *NMIs* and *Accuracies* than Sp*k*means in all the cases. Interestingly, there are many circumstances where Sp*k*means' result is worse than that of NMF clustering methods, but after refined by MVSCs, it becomes better.

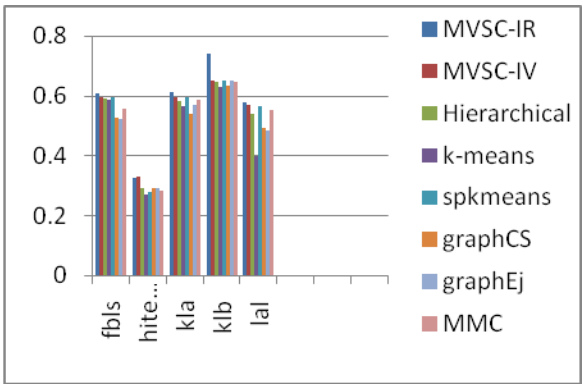

Fig 2: Clustering results in FScore

But they do not fare very well on the rest of the collections

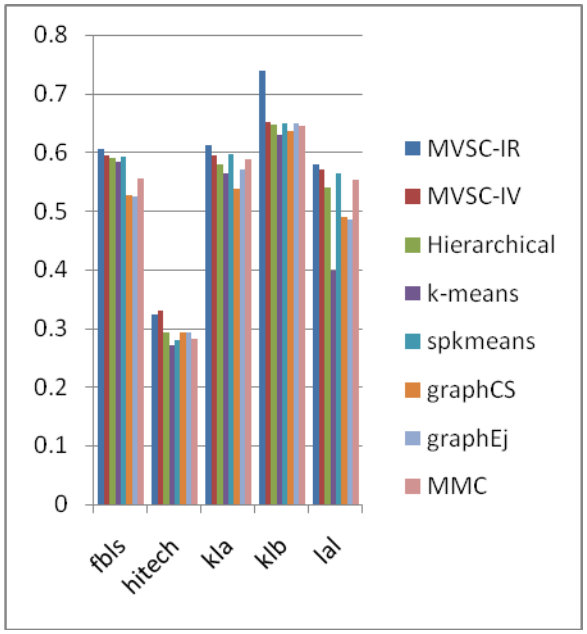

Fig 3: Clustering Results in NMI

V. CONCLUSION

In this paper we proposed a new similarity measure known as HMVS (Hierarchical Multi-Viewpoint based Similarity). When it is compared with cosine similarity, HMVS is more useful for finding the similarity of text documents. The empirical results and analysis revealed that the proposed scheme for similarity measure is efficient and it can be used in the real time applications in the text mining domain

References

[1] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, no. 1, pp. 1–37, 2007.

[2] I. Guyon, U. von Luxburg, and R. C. Williamson, "Clustering: Science or Art?" *NIPS'09 Workshop on Clustering Theory*, 2009.

[3] I. Dhillon and D. Modha, "Concept decompositions for large sparse text data using clustering," *Mach. Learn.*, vol. 42, no. 1 2,pp. 143–175, Jan 2001.

[4] S. Zhong, "Efficient online spherical K-means clustering," in *IEEE IJCNN*, 2005, pp. 3180–3185.

[5] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra, "Clustering on the unit hypersphere using von Mises-Fisher distributions," *J. Mach. Learn. Res.*, vol. 6, pp. 1345–1382, Sep 2005.

[6] W. Xu, X. Liu, and Y. Gong, "Document clustering based on nonnegative matrix factorization," in *SIGIR*, 2003, pp. 267–273.

[7] I. S. Dhillon, S. Mallela, and D. S. Modha, "Information-theoretic co-clustering," in *KDD*, 2003, pp. 89–98.

[8] C. D. Manning, P. Raghavan, and H. Sch ¨ utze, *An Introduction to Information Retrieval*. Press, Cambridge U., 2009.

[9] A. Strehl, J. Ghosh, and R. Mooney, "Impact of similarity measures on web-page clustering," in *Proc. of the 17th National Conf. on Artif. Intell.: Workshop of Artif. Intell. for Web Search*. AAAI, Jul. 2000, pp. 58–64.

[10] C. Ding, X. He, H. Zha, M. Gu, and H. Simon, "A min-max cut algorithm for graph partitioning and data clustering," in *IEEE ICDM*, 2001, pp. 107–114.

[11] Qin Ding, Maleq Khan, Amalendu Roy, and William Perrizo, "P-tree Algebra", ACM Symposium on AppliedComputing, Madrid, Spain, 2002.

[12] Maleq Khan, Qin Ding, William Perrizo, "K-Nearest Neighbor Classification of Spatial Data Streams using Ptrees",PAKDD-2002, Taipei, Taiwan, May 2002.

[13] Qin Ding, Qiang Ding, William Perrizo, "Association Rule Mining on Remotely Sensed Images using P-trees",PAKDD-2002, Taipei, Taiwan, 2002.

[14] Qin Ding, William Perrizo, Qiang Ding, "On Mining Satellite and other RSI Data", DMKD-2001, Santa Barbara, CA, 2001.

[15] A. Roy, "Implementation of Peano Count Tree and Fast P-tree Algebra", M. S. thesis, North Dakota State University, 2001.