International Journal of Computer Applications (0975 – 8887)
Volume 61– No.18, January 2013

Text Document Clustering based on Phrase Similarity using Affinity Propagation

Shailendra Kumar Shrivastava
Samrat Ashok Technological Institute Vidisha (MP) India

J.L.Rana, PhD.
Ex Head
Department C.S.E.
MANIT Bhopal, India

R.C.Jain, PhD.
Director
Samrat Ashok Technological Institute Vidisha (MP) India

ABSTRACT
Affinity propagation (AP) was recently introduced as an un-supervised learning algorithm for exemplar based clustering. In this paper novel text document clustering algorithm has been developed based on vector space model, phrases and affinity propagation clustering algorithm. Proposed algorithm can be called Phrase affinity clustering (PAC). PAC first finds the phrase by ukkonen suffix tree construction algorithm, second finds the vector space model using *tf-idf* weighting scheme of phrase. Third calculate the similarity matrix form VSD using cosine similarity .In Last affinity propagation algorithm generate the clusters .F-Measure ,Purity and Entropy of Proposed algorithm is better than GAHC ,ST-GAHC and ST-KNN on OHSUMED ,RCV1 and News group data sets.

Keywords
text clustering, affinity propagation, unsupervised learning, vector space model, suffix tree, tf-idf weighting scheme, Purity, Entropy-measure, GAHC, ST-GAHC, ST-KNN

1. INTRODUCTION
Clustering is fundamental task in computerized data analysis. It is concerned with the problem of partitioning a collection of data points into groups/categories using unsupervised learning techniques. Data points in groups are similar. Such groups are called clusters [1][2][3].Text document clustering algorithm attempt to group the documents based on their similarities, thus document relating to certain topic will hopefully be allocated into single cluster[4].Affinity propagation [5] is a clustering algorithm which for given set of similarities (also denoted by affinities) between pairs of data points, partitions the data by passing the messages among the data points. Each partition is associated with a prototypical point that best describes that cluster. AP associates each data point with one such prototype. Thus, the objective of AP is to maximize the overall sum of similarities between data points and their representatives. The literature founds two algorithms that uses affinity propagation algorithm in text clustering. Adaptive AP (AdAP) document clustering [6] [7] uses Vector Space Model (VSD) [8] to represent the document. VSD uses term frequency and Document frequency for weighting scheme. Seed affinity Propagation (SAP) [9] algorithm is based on semi-supervised learning. Disadvantage of SAP is that it required labeled information for seed construction. Both algorithm uses the word base VSD hence result are not good.
Zamir et al. [10] and Zamir and Etzioni [11] had proposed Phrase based document similarity is in Suffix Tree Clustering (STC).STC was based on STD model[12] . The STC algorithm got poor results in clustering the documents in their experimental data sets of RCV1 data set [13].
STD model is based on phrase but the clustering algorithm based on STD model are not good because STD model in not using the term weighting scheme.VSD model is based on word hence the results of clustering algorithm based on VSD model are not good. Chim et al [14] and Chim et al.[15] combine two model to take the benefits of both models. Weiner [16], Mcreight [17] and Esko Ukkonen[18] have given their on suffix tree construction algorithm . Esko Ukkonen suffix tree construction algorithm has time complexity O(N) less than all the suffix tree construction algorithm. Hence ,Esko Ukkonen suffix tree construction algorithm for getting the phrases has been used.

Phrase based vector space model has been used in proposed algorithm. It construct the suffix tree using Esko Ukkonen algorithm. Extract phrase form tree from the suffix tree . Next it find the *phrase* frequency and *inverse* phrase frequency and construct the VSD. The Literature survey find that Euclidian similarity [1], Jaccard similarity, and Cosine similarity are used to find the similarities among documents. Guan el al [9] ,and Chim et al[15] found the results with cosine similarity are good. Hence weused the cosine similarity in proposed algorithm. In last clusters will be generated by Affinity clustering algorithm.
The remainder of this paper is organized as follows. Section 2 gives a brief over view of original Affinity Propagation algorithm, Adaptive AP document clustering, Seed affinity Propagation (SAP) Section 3 introduces the main idea and details of proposed algorithm. Section 4 discusses the experimental results and evaluation. Section 5 provides the concluding remarks and future directions.

2. RELATED WORKS
Before going into details of proposed novel PAC algorithm, some works that are closely related to this paper are briefly reviewed.Adaptive AP document clustering, Seed affinity Propagation (SAP) , Vector Space Model , Suffix Treewill be discuss.For the sake of continuity affinity propagation algorithm will be discuses first.

2.1 Affinity Clustering Algorithm
Affinity clustering algorithm [5] is based on message passing among data points. Each data point receives the availability from others data points (from exemplar) and send the responsibility message to others data points (to exemplar).Sum of responsibilities and availabilities for data points identify the exemplars. After the identification of exemplar the data points are assigned to exemplar to form the clusters. Following are the steps of affinity clustering algorithms.

1. Initialize the availabilities to zero $a(i,k)=0$
2. Update the responsibilities by following equation.
$r(i,k) \longleftarrow (s(i,k) + \max_{k'st\ k' \neq k}\{a(i,k' + s(i,k')\}$(1)

Where $s(i,k)$ is the similarity of data point *i* and exemplar *k*.

3. Update the availabilities by following equation
$a(i,k) \leftarrow$
$min\{0, r(k,k) + \sum_{i' s.t. i' \notin \{i,k\}} max\{0, r(i',k)\}\}$(2)

4. Update self-availability by following equation
$a(k,k) \longleftarrow \sum \max(\{0, r(i',k)\})$ (3)

5. Compute sum = $a(i,k) + r(i,k)$for data point i and find the value of k that maximize the sum to identify the exemplars.

6. If Exemplars do not change for fixed number of iterations go to step (6) else go to
Step (1)

Assign the data points to Exemplars on the basis of maximum similarity to find clusters.

2.2 Adaptive AP (AdAP) document clustering

He et al had proposed adaptive AP (AdAP) [6][7] for document clustering. AdAP for document clustering uses Vector Space Model (VSD) to represent the document. VSD uses term frequency and Document frequency of words in weighting scheme. In this document representation AdAP is applied to find the clusters.

2.3 Seed affinity Propagation (SAP)

Guan et al. had proposed a Seed affinity Propagation (SAP) [9] for text clustering. This algorithm is based on semi-supervised learning. The Labeled information is available (Title of Document). Labeled information is included with unlabeled information this is known as seed construction. Compute the Tri-Features set (UFS, CFS and SCFS).InTri-set computation include the textural information. Calculate the similarity matrix using these three features sets. Now Apply AP steps to find the text clusters. The Results of SAP is better than K-means and AP.

2.4 Vector Space Model

Vector space model [8]use to represent the text documents. In VSD each document *d* is considered as a vector in the M-dimensional term (word) space. PAC used the *tf-idf*weighing scheme .In VSD model each document represented by following equation.

$\vec{d} = \{w(1,d), w(2,d), \ldots . w(N,d)\}$(4)

Where N is the number of term (word) in the document.And
$w(i,d) = (1 + \log\, tf(i,d) * \log(1 + N/df(i))$ (5)

where $tf(i,d)$ frequency of ith term in the document *d* and *df*(*i*) is the number of document containing ith term . Inverse document frequency (*idf*) is defined as the logarithm of the ratio of number of documents (N) to the number of document containing the given word (*df*)

2.5 Suffix Tree

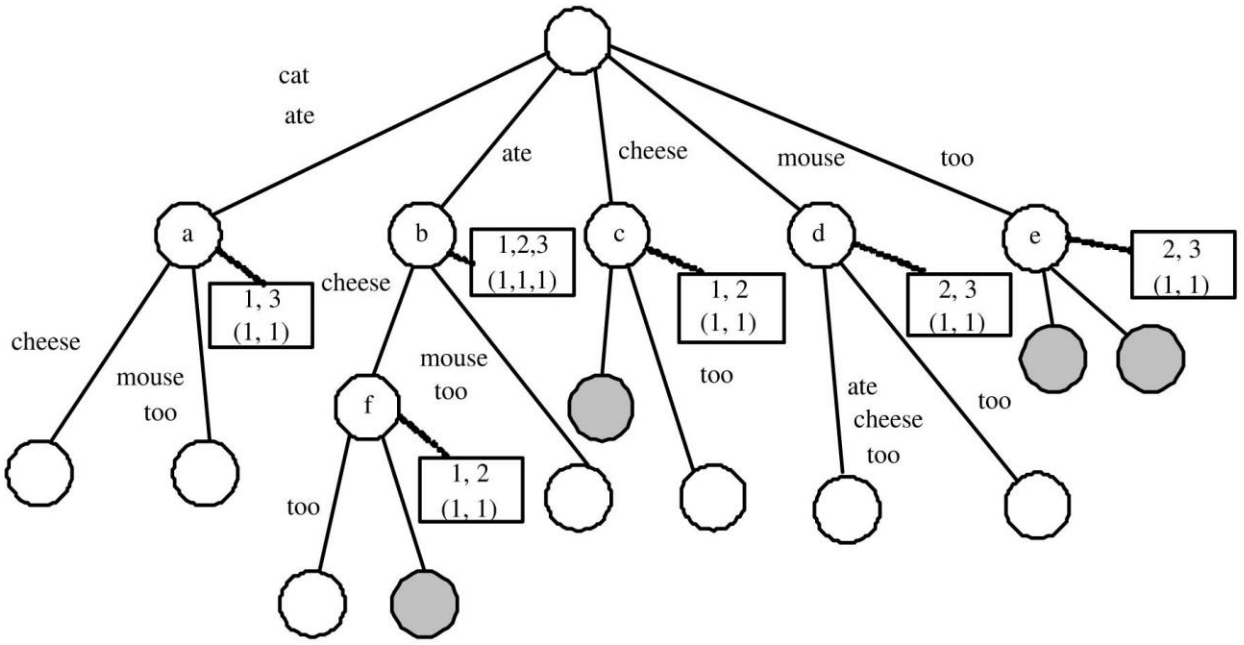

Fig 1: Suffix tree of tree documents" cat ate cheese," mouse at Cheese too" and "cat ate mouse too" [15]

Suffix Tree[12][19]is a tree like data structure representing all suffix substrings of a string. Suffix tree document model represent document as string consisting the sequence of words. Construct the suffix tree of document by ukkonen [18] algorithm. Suffix tree has root node, internal node, leaf node and null node. Internal node has at least two children. Each edge in suffix tree is labeled with non-empty substring of as document called phrase. Leaf node in the suffix tree represents the substring of document. Each internal node represents a common phrase shared by at least two suffix substrings of document. Similarity of documents defined as more nodes shared by documents. Null node represent nothing but they generated during construction of tree by ukkonen algorithm In Figure 1, each internal node is attached to individual box . The numbers in box represent the document that has traversed the corresponding node. Each upper number represent the document identifier and number below represent the traversed times of document. Ukkonen algorithm constructs a series of implicitsuffix trees. An implicit suffix tree is a simplified suffix tree without a terminating character to the string, with labeled edges and all nodes have at least two children. Ukkonen algorithm is divided into *m* phases.In phase i+1, tree *Ii*+1is constructed from *Ii*.Each phase i+1 is further divided into *i*+1 extension one for each of the *i*+1 suffixes of S [1…*i*+1]. In extension jof phase *i* + 1, the algorithm first finds the end of the path from the root labeled with substring S [*j*...*i*]. It then extends the substring by adding the character S (*i* + 1) to its end, unless S (*i* + *I*) already appears there.So in phase i + I, string S[1..*i* + 1] is first put in the tree, followed by strings S[2..*i* +1], S[3..*i*+1],… (In extensions 1, 2, 3, , respectively). Extension *i* + 1 of phase *i* + 1 extends the empty suffix of S[1..*i*],that is , it puts the single character string S(*i* + 1) into the tree (Unless it is already there). Tree *Ii* is just the single edge labeled by character S(1). Let S[*j*..*i*l = β be a suffix of S [1...*i*]. In extension *j*, when the algorithm finds the end of β in the current tree, it extends β to be sure the suffix β S(i + 1) is in the tree. It does this according to one of the following three Suffix extension rules.

Rule 1

In the current tree, path β ends at a leaf. That is, the path from the root labeled β extends to the end of some leaf edge. To update the tree, character S (*i* + 1) is added to the end of the label on that leaf edge.

Rule 2

No path from the end of string β starts with character S(i + 1), but at least one labeled path continues from the end of β.In this case, a new leaf edge starting from the end of β must be created and labeled with character S(i + 1). A new node will also have to be created there if, β ends inside an edge. The leaf at the end of the new leaf edge is given the number j.

Rule 3

Some path from the end of string β starts with character S(i +1). In this case the string βS (i + 1) is already in the current tree, so (remembering that in an implicit suffix tree the end of a suffix need not be explicitly marked) we do nothing.

3. PROPOSED PHRASE AFFINITY CLUSTERING (PAC)

In novel proposed algorithm for Text Document Clustering Based On Phrase Similarity Using Affinity Propagation has benefits of STD Model and Vector Space model and affinity propagation.

In vector space model [8] *w* (*d*, *i*), term frequency and document frequency is calculated on the basis of term. Term in vector space model is word .Algorithm uses phrase instead of word. This can be called Vector space model bases on the phrase. Phrase (Term) frequency and document frequency can be calculated by suffix tree. Here document frequency is number document contains the phrase. VSD model based on the phrase is used to compute the cosine similarity. Similarity of two document $d_i$and d_j is calculated by following formula and document can be represented by equation (4).

Next, finds the cosine similarity [1] by equation 6 .

$$sim(d_i , d_j) = \frac{\overrightarrow{d_\iota} \bullet \overrightarrow{d_J}}{|d_i|x|d_j|} = \frac{\sum_i^N d_i\, d_j}{\sqrt{\sum_i^N d_i^2 \sum_i^N d_j^2}} (6)$$

Self-similarity/Preference [9] is finding from by equation

$$7sim(d_k , d_k) = \frac{\sum_{i,j=1,i\neq j}^{N} sim(d_i ,d_j)}{N\times N(N-1)} \quad 1 \le k \le N(7)$$

Now affinity propagation algorithm for clustering is applied to generate the clusters. Proposed algorithm can be written as following.

1. Input Text for Clustering
2. Document preprocessing.
   Removing all stop words.
   Words steaming are done.
3. Find the words and assign the unique number to each word.
4. Convert text into sequence of number.
5. Suffix tree construction using Ukkonen algorithm.
6. Calculate the Phrase (term) frequency from suffix tree.
7. Calculate the document frequency of phrase from suffix tree.
8. Construct the Vector space model of text using phrase.
9. Find the Phrase based similarity matrix of documents from vector space model by equation(1).
10. Preference in similarity matrix is assigned by equation (2).
11. Initialize the availabilities to zero $a(i, k) = 0$
12. Update the responsibilities by following equation.
13. Update the availabilities by following equation
14. Update self-availability by following equation
15. Compute sum = $a(i, k) + r(i, k)$ for data point i and find the value of k that maximize the sum to identify the exemplars.
16. If Exemplars do not change for fixed number of iterations go to step (6) else go to Step (1)

Assign the data points to Exemplars on the basis of maximum similarity to find clusters.

4. EXPERIMENTAL RESULTS AND EVALUATION

In this Section, results and evaluation of set of experiments are presented to verify the effectiveness and efficiency of proposed algorithm for clustering. Evaluations parameters are F-Measures, Purity and Entropy. Experiments have been performed on data sets constructed from Corpus OHSUMED[15], RCV1[13] and Newsgroup[15]. We will discuss Evaluation parameter, Dataset description and results.

4.1 Evaluations parameters[9]

For ready reference definition and formulas of F-Measure, Purity and Entropy are given below..

4.1.1 F-measure

F-Measure combines the Precision and Recall. Let C={$C_1 C_2 C_3$ ... C_k } be clusters of data set D of N documents ,and let $C^* = C_1^*, C_2^*$ ... C_l^*represents the correct class of set D.Then the Recall of Cluster *j* with respect to Class *i* is defined as

$$Recall(i , j)= \frac{|C_j \cap C_j^*|}{|C_i^*|}$$

Then the Precision of Cluster *j* with respect to Class *i* is defined as

$$Precision (i , j)= \frac{|C_j \cap C_j^*|}{|C_j|}$$

F-Measures of cluster C_j and class C_i^* is the combinations of Precision and Recall in following manner.

$$F(i,j) = \frac{2 * Precision\ (i ,j) * \ Recall(i ,j)}{Precision\ (i ,j) + Recall(i ,j)}$$

F-Measure for overall quality of cluster set C is defined by the following equation

$$F = \sum_{i=1}^{l} \frac{|C_i^*|}{N} * \{\max_{j=1,..,l} F(i,j)\}$$

4.1.2 Purity

Purity indicates the percentage of the dominant class member in the given cluster. For measuring the overall clustering purity weighted average purity is used. Purity is given by following equation.

$$Purity\ = \sum_{i=1}^{k} \frac{|C_j|}{N} * \{\max_{j=1,..,l} Precision\ (i\ ,j\ )\}$$

4.1.3 Entropy

Entropy tell us the homogeneity of cluster. Higher homogeneity of cluster entropy should be low and vice versa. Like weighted F-Measure and weighted Purity weighted entropy is used which is given by following equation

$$Entropy = -\frac{1}{\log k} \sum_{j=1}^{k} \frac{|C_j|}{N} \sum_{i=1}^{l} p_{ij} \log p_{ij}$$

Where $p_{ij}$is the probability that a member of cluster C_j belongs to class C_i^* .

To sum up, we would like to maximize the F-Measure and Purity scores and minimize the entropy score of cluster to achieve high quality clustering.

4.2 Data Set Preparation

Data sets DS1,DS2,DS3,DS4,DS5,DS6 and DS7 are generated to perform experiments. DS1,DS2 and DS3 are the data sets generated by OHSUMED corpus[20] , data sets. DS4,DS5 and DS6 generated from RCV1 corpus [13] and data sets and DS7 from 20-News Groups[15] .Data sets generated by approach follow by Chim et al.[14][15] Table shows the data sets used in evaluation of proposed algorithm.

Table 1. Overview of the Document Sets [15]
(Corpus Type: O—OHSUMED, R—RCV1 ,N-Newsgroup)

Document Set	**DS1**	**DS2**	**DS3**	**DS4**	**DS5**	**DS6**	**DS7**
Corpus Type	O	O	O	R	R	R	N
Categories	3	5	8	4	6	10	20
Documents	300	500	800	400	600	2000	2000

4.3 Experimental Result Discussion

Figures 1,2,3,4 respectively and Table 2, 3, 4 respectively illustrate the *F-Measure, Purity* ,and *Entropy* scores computed from the clustering results of four clustering algorithms on seven document data sets. Proposed novel Phrase Affinity clustering is implemented in MATLAB 11 and obtained results are shown in graphs and tables. Results of Group Average Hierarchical Clustering (GAHC), Suffix Tree Group Average Hierarchical Clustering (ST-GAHC), Suffix Tree K-Nearest Neighbors (ST-KNN) were obtained from research paper of chim el al[15]. Because data sets are designed from same logic and same corpus. On the basis of figure 1,2,3,4 and Table 2,3,4 it can observed that F-Measure, Purity and Entropy of Proposed Phrase Affinity is outperform GAHC, ST-GAHC, ST-KNN.

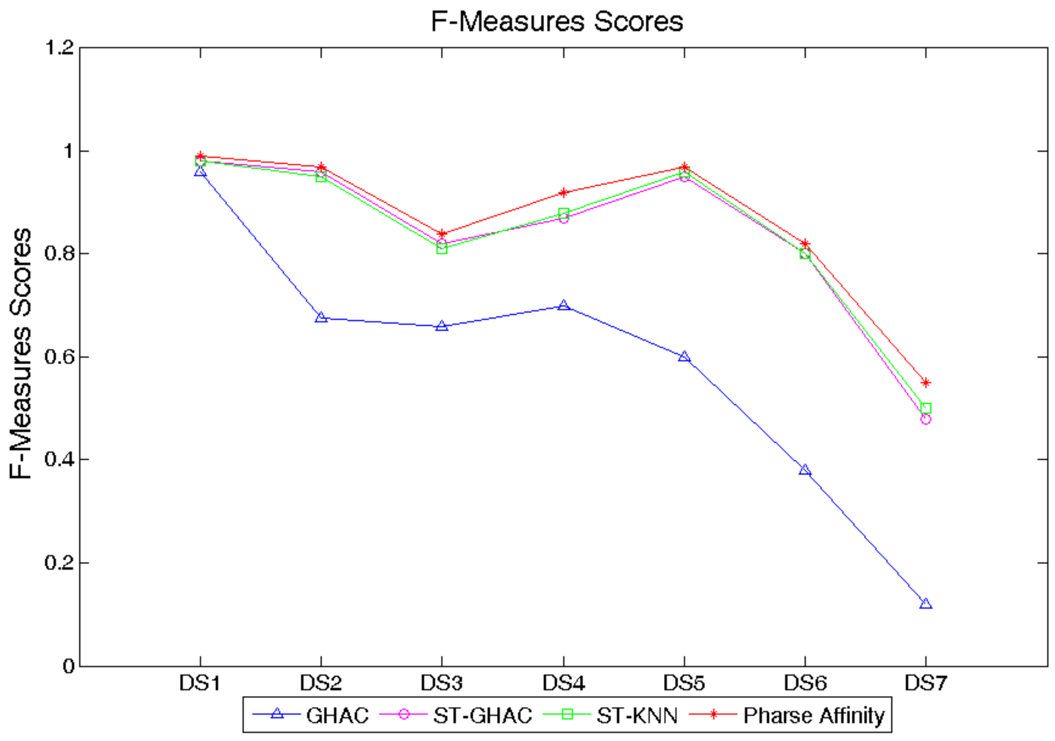
F-Measures Scores
F-Measures Scores
1.2
1
0.8
0.6
0.4
0.2
0
DS1
DS2
DS3
DS4
DS5
DS6
DS7
GHAC
ST-GHAC
ST-KNN
Pharse Affinity

Fig 2 :

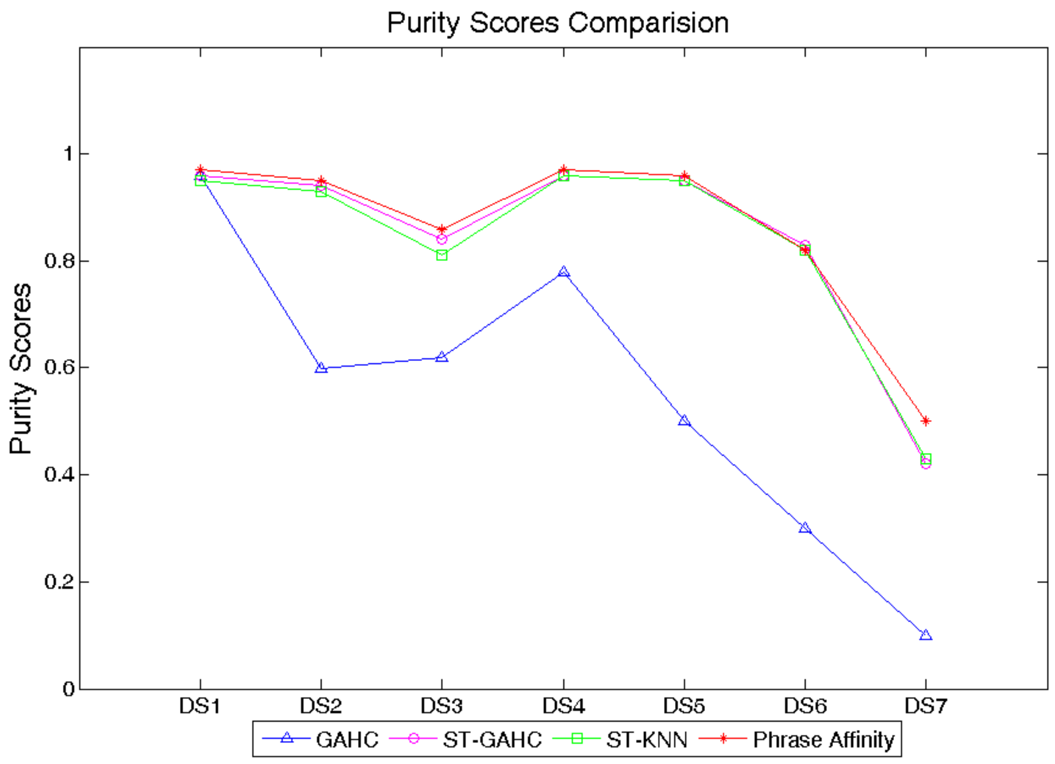
Purity Scores Comparision
Purity Scores
1
0.8
0.6
0.4
0.2
0
DS1
DS2
DS3
DS4
DS5
DS6
DS7
GAHC
ST-GAHC
ST-KNN
Phrase Affinity

Fig 3 :

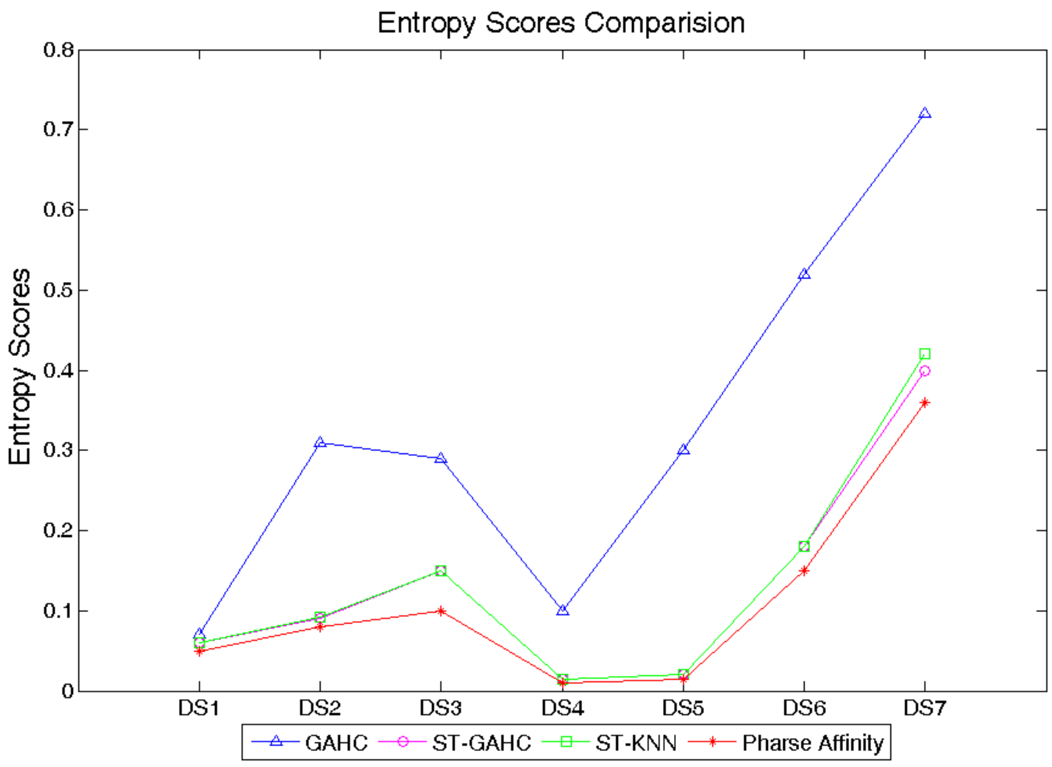

Fig 4 :

Table 2. Comparison of F-measure scores

Data Sets/Algorithms	GAHC	ST-GAHC	ST-KNN	Proposed Novel Phrase Affinity Clustering
DS1	0.96	0.98	0.98	0.99
DS2	0.675	0.96	0.95	0.97
DS3	0.66	0.82	0.81	0.84
DS4	0.7	0.87	0.88	0.92
DS5	0.6	0.95	0.96	0.97
DS6	0.38	0.8	0.8	0.82
DS7	0.12	0.48	0.5	0.55

Table 3. Comparison of Purity

Data Sets/Algorithms	GHAC	ST-GHAC	ST-KNN	Proposed Novel Phrase Affinity Clustering
DS1	0.96	0.96	0.95	0.97
DS2	0.6	0.94	0.93	0.95
DS3	0.62	0.84	0.81	0.86
DS4	0.78	0.96	0.96	0.97
DS5	0.5	0.95	0.95	0.96
DS6	0.3	0.83	0.82	0.82
DS7	0.1	0.42	0.43	0.5

Table 4. Comparison of Entropy

Data Sets/Algorithms	GHAC	ST-GHAC	ST-KNN	Proposed Novel Phrase Affinity Clustering
DS1	0.07	0.06	0.06	0.05
DS2	0.31	0.09	0.092	.08
DS3	0.29	0.15	0.15	0.10
DS4	0.1	0.015	0.015	0.01
DS5	0.3	0.02	0.02	0.015
DS6	0.52	0.18	0.18	0.15
DS7	0.72	0.39	0.42	0.36

5. CONCLUDING REMARKS AND FUTURE DIRECTIONS

Proposed Novel Phrase affinity clustering (PAC) algorithm is based on Phrases, Vector space model, tf-*idf* weighting scheme and affinity clustering. Hence F-Measures, Purity and Entropy of PAC is better than GAHC, ST-GAHC, ST-KNN. Extensive experiments on many standard datasets show that the proposed PAC algorithm produces clusters of better clustering accuracy.

There are a number of interesting potential avenues for future research. PAC can be applied in information retrieval, image segmentation. Extensive Experiments can be carried out to show the effectiveness of PAC for large data sets. PAC execution speed can be increase by fixing minimum phrase frequency and document frequency in term of phrase. Fast affinity clustering based on machine learning can be applied Phrase based similarity matrix.

6. REFERENCES

[1] RuiXu Donald C. Winch, "Clustering" , IEEE Press 2009 ,pp 1-282

[2] Jain, A. and Dubes R. "Algorithms for Clustering Data ", Englewood Cliffs, NJ Prentice Hall, 1988.

[3] A.K. Jain, M.N. Murthy and P.J. Flynn, "Data Clustering: A Review ", ACM Computing Surveys, Vol.31. No 3, September 1999, pp 264-322.

[4] RuiXu, and Donald Wunsch," Survey of Clustering. Algorithms ", IEEE Transactions on Neural Network, Vol 16, No. 3, 2005 pp 645.

[5] Frey, B.J. and DueckD." Clustering by Passing Messages Between Data Points ", Science 2007, pp 972–976.

[6] Kaijun Wang, Junying Zhang, Dan Li, Xinna Zhangand Tao Guo, "Adaptive Affinity Propagation Clustering",ActaAutomaticaSinica, 2007 ,1242-1246.

[7] Yancheng He , Qingcai Chen, Xiaolong Wang, Ruifeng Xu, Xiaohua Bai and Xianjun Meng ," An Adaptive Affinity Propagation Document Clustering", 7th International Conference on Informatics and Systems (INFOS), 2010,pp 1-7.

[8] Salton G., Wong A., and Yang C. S., 1975, "A Vector Space Model for Automatic Indexing," Comm. ACM, vol. 18, no. 11, pp. 613-620.

[9] Renchu Guan, Xiaohu Shi, Maurizio Marchese, Chen Yang, and Yanchun Liang," Text Clustering with Seeds Affinity Propagation ",IEEE Transaction on Knowledge and Data Engineering Vol. 23 No 4,2011,pp 627-637

[10] O.M. Oren Zamir, O. Etzioni, and R.M. Karp, "Fast and Intuitive Clustering of Web Documents," Proc. Third Int'l Conf. Knowledge Discovery and Data Mining (KDD), 1997.

[11] Oren Zamir and Oren Etzioni. Grouper: a dynamic clustering interface to Web search results. Computer Networks (Amsterdam, Netherlands: 1999.

[12] Sven Meyer D. S., Eissen Zu. and Potthast M., 2005, "The Suffix Tree Document Model Revisited," Proc. Fifth Int'l Conf. Knowledge Management (I-Know '05), pp. 596-603.

[13] D.D. Lewis, Y. Yang, and F. Li, "RCV1: A New Benchmark Collection for Text Categorization Research," J. Machine Learning Research, vol. 5, pp. 361-397, 2004.

[14] Hung Chim ,Xiaotie Deng ,"A New Suffix Tree Similarity Measure for Document",Proceedings of the 16th international conference on World Wide Web ACM New York, NY, USA ,2007 ,Pages 121-130

[15] Chim H. and Deng X., 2008 "Efficient Phrase Based Document Similarity for Clustering", IEEE Trans. Knowledge and Data Engineering, vol. 20, No.9.

[16] P. Weiner. Fast and effective text mining using linear-time document clustering. In Proceedings of 14th Annual IEEE Symposium on Switching and Automata Theory, The University of lowa, 1973.

[17] Edward M. McCreight. A space-economical Suffix tree construction algorithm. Journal of ACM, 1976. Page 154

[18] Esko Ukkonen. On-line construction of suffix trees. Algorithmica, 1995,Page 260.

[19] Sven Meyer D. S., Eissen Zu. and Potthast M., 2005, "The Suffix Tree Document Model Revisited," Proc. Fifth Int'l Conf. Knowledge Management (I-Know '05), pp. 596-603.

[20] Hersh W., Buckley C., and Hickam D., 1994, "Ohsumed: An Interactive Retrieval Evaluation and New Large Test Collection for Research," Proc. 17th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '94), pp. 192-201