Managing Email Overload with an Automatic Nonparametric Clustering Approach

Yang Xiang[1], Wanlei Zhou[2], and Jinjun Chen[3]

[1] School of Management and Information Systems, Central Queensland University
Rockhampton, Queensland 4700, Australia
y.xiang@cqu.edu.au
[2] School of Engineering and Information Technology, Deakin University
Burwood, Victoria 3125, Australia
wanlei@deakin.edu.au
[3] Faculty of Information & Communication Technologies, Swinburne University of Technology, Hawthorn 3122, Australia
jchen@ict.swin.edu.au

Abstract. Email overload is a recent problem that there is increasingly difficulty people have faced to process the large number of emails received daily. Currently this problem becomes more and more serious and it has already affected the normal usage of email as a knowledge management tool. It has been recognized that categorizing emails into meaningful groups can greatly save cognitive load to process emails and thus this is an effective way to manage email overload problem. However, most current approaches still require significant human input when categorizing emails. In this paper we develop an automatic email clustering system, underpinned by a new nonparametric text clustering algorithm. This system does not require any predefined input parameters and can automatically generate meaningful email clusters. Experiments show our new algorithm outperforms existing text clustering algorithms with higher efficiency in terms of computational time and clustering quality measured by different gauges.

Keywords: Email, overload, text clustering, knowledge management.

1 Introduction

As part of daily life, email has made significant changes to the way of exchanging and storing information. According to the estimate in [1], the number of worldwide email messages sent daily has reached 84 billion in 2006. On one side, email can be an effective knowledge management tool that conveniently enables fast and accurate communication. On the other side, the increasing volume of email threatens to cause a state of "email overload" [2] where the volume of messages exceeds individuals' capacity to process them. This is because of the fact that the majority of email users use email as an archival tool and never discard messages [3]. As this gradual congestion of a user's mailbox with messages ranging from working related documents and personal information, users are becoming unable to successfully

K. Li et al. (Eds.): NPC 2007, LNCS 4672, pp. 81–90, 2007.

finding an important archived message hidden in their mailbox without any structure. We urgently need an effective managing tool to solve the email overload problem.

Recently people realize that categorizing email messages into different groups can significantly reduce the cognitive load on email users [4]. If an email system can present messages in a form that is consistent with the way people process and store information, it will greatly help them in comprehending and retrieving the information contained within those groups [5]. Currently categorizing email messages often involves manually creating folders in users' mailbox and setting up rules to dispatch incoming emails [6, 7]. This requires heavy cognitive load of creating the folder structure and the rules, which can be difficult for normal users.

In this paper, we propose a new automatic nonparametric clustering approach to manage email overload. We implement this system with a client-side prototype application. It can automatically generate email clusters according to emails' content (title and body) by a new text clustering algorithm. It does not require users to input any predefined parameter and therefore it is especially useful for non-technical users. The evaluation shows our approach achieves good clustering results with high efficiency and high clustering quality.

2 System Design

The system design goal of the automatic email clustering system is to automatically categorize emails into different meaningful groups and thus to alleviate human cognitive load to process emails. The emails in clusters the system produced must be of same relative group. We also notice a fact that most actual email users manually create 1-level folders in their mail box, rather than create multi-level hierarchical folders because exploring the multi-level hierarchical structure is also burdensome. Therefore, we choose 1-level folders containing emails as the output in this system. We implement the system by a client-side prototype application. It first read email messages from an email client's data file, then it converts email texts into vector matrix and generate similarity matrix. The details regarding algorithms will be introduced in section 3. After the matrices are generated, they are input into our new nonparametric text clustering algorithm. The algorithm produces email clusters. Finally the application outputs 1-level email clusters in a user interface. The flow chart can be found in figure 1.

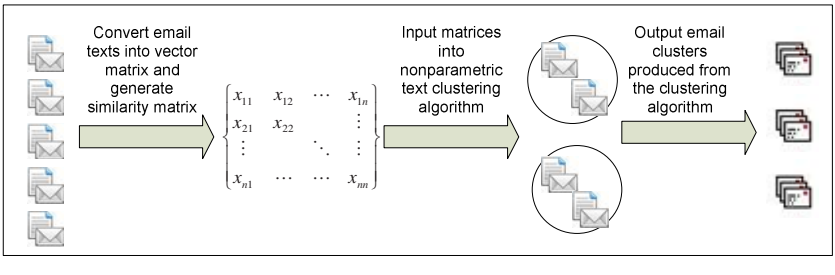

Fig. 1. System flow chart

3 A New Nonparametric Text Clustering Algorithm

3.1 Cluster Validity

To validate a cluster analysis solution, there are many validation techniques such as Hubert's Γ statistic [8, 9], significance tests on variables (MANOVA) [10], and Monte Carlo procedures [9]. We use Hubert's Γ statistic which does not require predefined parameters to validate the cluster structures. The underlying idea is that our algorithm aims at evaluating the degree of agreement between a predetermined partition and the inherent clustering structure. Therefore, our algorithm combines the cluster optimization and clustering itself together, which produces high quality clusters and achieve high efficiency.

First let us define an $N \times N$ proximity matrix $\mathrm{Y} = [Y(i, j)]$, where

$$Y(i, j) = \begin{cases} 1, \text{if emails } i \text{ and } j \text{ are clustered in the same cluster} \\ 0, \text{if emails } i \text{ and } j \text{ are not clustered in the same cluster} \end{cases} \tag{1}$$

Consistently, the inherent clustering structure can be represented by using an $N \times N$ proximity matrix $\mathrm{X} = [X(i, j)]$, to observe correlation coefficient of emails i and j, which means the proximity of point i to point j in the whole email space.

The Hubert's Γ statistic then can be obtained by measuring the correlation between these two symmetric matrices $\mathrm{X} = [X(i, j)]$ and $\mathrm{Y} = [Y(i, j)]$ as follows

$$\Gamma = \frac{1}{n(n-1)/2} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(\frac{X(i, j) - \overline{X}}{\sigma_X} \right) \left(\frac{Y(i, j) - \overline{Y}}{\sigma_Y} \right) \tag{2}$$

Where $\overline{X}$ and $\overline{Y}$ are the means of the values in X and Y, and σ_X and σ_Y are the standard deviations. The value of Hubert's Γ statistic scaled from -1 to 1, and a large absolute value of the normalized statistic implies well separated and compact clusters. (2) can be further written as

$$\begin{aligned} \Gamma &= \frac{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(X(i, j) - \overline{X} \right) \left(Y(i, j) - \overline{Y} \right)}{\sqrt{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(X(i, j) - \overline{X} \right)^2} \sqrt{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(Y(i, j) - \overline{Y} \right)^2}} \\ &= \frac{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(X(i, j) Y(i, j) - \overline{X} Y(i, j) - X(i, j) \overline{Y} + \overline{X} \overline{Y} \right)}{\sqrt{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(X(i, j) - \overline{X} \right)^2} \sqrt{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(Y(i, j) - \overline{Y} \right)^2}} \end{aligned} \tag{3}$$

The Hubert's Γ statistic has twofold meanings. First, the proximity matrices X and Y measures are generic to cover both inter-class and intra-class statistics. Second, when the product of $X(i, j)$ and $Y(i, j)$ is large, it is very likely that the email points are apart and assigned to different clusters with distant mean vectors. Therefore we can see that the larger the Hubert's Γ statistic means stronger evidence there are compact

clusters generated. To this end, we use the maximum Hubert's Γ statistic as our cluster validation measure as follows

$$Validity = \arg(\max(\Gamma)) \tag{4}$$

3.2 Vector Space Model for Text Clustering

Before emails can be classified into different clusters, they must be represented by a numerical form. In this paper we adopt the vector space model [11], which has been used in Information Retrieval to compute the degree of similarity between each text document stored in the system. In this model, each email is converted to a vector e, in the term space, as follows.

$$e = \{tf_1, \cdots, tf_i, \cdots, tf_K\} \tag{5}$$

Where tf_i is the frequency of the ith term in the email. In the vector space model, Intra-similarity is quantified by measuring the raw frequency of a term t_i inside a email document e_i. tf_i is usually normalized as formula (6) to prevent a bias towards longer emails (which may have a higher term frequency regardless of the actual importance of that term in the email) to give a measurement of the importance of the term t_i within the particular email.

$$tf_i = \frac{n_i}{\sum_k n_k} \tag{6}$$

Where n_i is the number of emails in which the index term t_i appears, and the denominator is the number of occurrences of all terms.

Inter-cluster dissimilarity is quantified by measuring the inverse of the frequency of a term t_i among the emails in the whole space. This factor is usually referred as the inverse document frequency or the idf factor. The motivation for usage of an idf factor is that terms which appear in many emails are not necessarily useful for distinguishing a relevant email from non-relevant ones. idf can be written as

$$idf_i = \log \frac{N}{n_i} \tag{7}$$

Where N is the total number of emails in the system. Then $tfidf$ can be used to filter out common terms which have little discriminating power, as defined in the follows.

$$ifidf = tf_i \times idf_i \tag{8}$$

As our clustering algorithm has many iteration steps, in each iteration step the goal is to separate the existing collection of emails into two sets: the first one that is composed of emails related to the currently generated cluster and the second one is composed of emails not related it. Two main issues need to be resolved, intra-cluster

similarity and inter-cluster dissimilarity. The quantification of intra-cluster similarity provides the features which better describe the emails in the currently generated cluster. Furthermore, the quantification of inter-cluster dissimilarity represents the features which better distinguish the emails in currently generated cluster. Therefore, intra-similarity is quantified by measuring the term frequency tf_i. Inter-cluster dissimilarity is quantified by measuring the inverse of the frequency idf_i.

3.3 Nonparametric Text Clustering Algorithm

Our new nonparametric text clustering algorithm has the following 7 steps.

1. Construct the data matrix with the vector space model;
2. Standardize the data matrix;
3. Compute the similarity matrix and input similarity matrix into the clustering procedure;
4. Select a seed for clustering;
5. Add or delete point into the currently generated cluster with Hubert's Γ statistic validity test;
6. Repeat step 5 until no point can be allocated;
7. Repeat step 4 to 6 until all the clusters are generated.

The key differences between our algorithm and traditional clustering algorithms such as hierarchical agglomerative algorithm and partitioning-based k-means algorithm [9] are first, there is no need to reconstruct the data or similarity matrix for each iteration, which can greatly save computational time; and second, it incorporates the validation part into to the clustering process, instead of putting the validation after all clusters are generated, which can optimize the quality of clustering all the time.

The clustering result does not depend on the selection of seed in step 4, although a good selection can speed up the clustering process. We use Euclidean distance measure to choose the seed, which has most neighbour points with the average Euclidean distances among points. The Euclidean distance measurement can be written as

$$d_{ij} = \left\{ \sum_{k=1}^{p} \left(X_{ik} - X_{jk} \right)^2 \right\}^{\frac{1}{2}} \tag{9}$$

Where X_{ik} is the value of the *k*th variable for the *i*th point. The average Euclidean distances then can be written as

$$\bar{d} = \frac{1}{n(n-1)/2} \sum_{i=1}^{n-1} \sum_{j=i}^{n} d_{ij} \tag{10}$$

3.4 Computational Simplification

The Hubert's statistic is robust because the inter-class and intra-class information is embedded into cluster evaluation. However, the original Hubert's statistic approach

requires high computation load. Therefore, we make further simplification for (3) as follows.

$$\Gamma = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\left(X(i,j)Y(i,j) - \overline{X}Y(i,j) - X(i,j)\overline{Y} + \overline{X}\overline{Y}\right)}{\sqrt{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\left(X(i,j)-\overline{X}\right)^2}\sqrt{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\left(Y(i,j)-\overline{Y}\right)^2}}$$

$$= \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)Y(i,j) - \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)}{n(n-1)/2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j) - \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)}{n(n-1)/2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j) + \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)}{n(n-1)/2}}{\sqrt{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\left(X(i,j) - \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)}{n(n-1)/2}\right)^2}\sqrt{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\left(Y(i,j) - \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)}{n(n-1)/2}\right)^2}} \tag{11}$$

$$= \frac{\left(\frac{n(n-1)}{2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)Y(i,j) - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)\right)}{\sqrt{\frac{n(n-1)}{2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)^2 - \left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)\right)^2}\sqrt{\frac{n(n-1)}{2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)^2 - \left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)\right)^2}}$$

From this formula we find many parts can be pre-calculated when the data and similarity matrix are given, which is before the iteration (from step 4 to 6) starts. If we can pre-calculate these parts, which are defined as follows, the computational time can be greatly saved.

$$A = \frac{n(n-1)}{2} \tag{12}$$

$$B = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j) \tag{13}$$

$$C = \sqrt{\frac{n(n-1)}{2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)^2 - \left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)\right)^2} \tag{14}$$

Then by substituting (12)(13)(14) into (11) we have

$$\Gamma = \frac{A\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}X(i,j)Y(i,j) - B\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)}{C\sqrt{\frac{n(n-1)}{2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)^2 - \left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}Y(i,j)\right)^2}} \tag{15}$$

Therefore the actual computation needed in each iteration is for those parts with $Y(i,j)$.

4 Evaluation

4.1 Measurements

The performance of a text clustering algorithm can be evaluated in terms of computational time, and quality measure (high intra-cluster similarity and low inter-cluster

similarity). We first use some unlabelled email data sets to test our algorithm on the first two measurements. We also compare our result with other two clustering algorithms, hierarchical agglomerative algorithm (using the nearest neighbour algorithm) and k-means algorithm. The quality of clustering is measured by using different measurements, Hubert's Γ statistic, simple matching coefficient, and Jaccard coefficient. Readers can find the definitions of simple matching coefficient and Jaccard coefficient in [9]. A higher value in these measurements indicates a higher clustering quality.

4.2 Data Sets

All the email data sets are from real life email collections in a university environment. Stop words are removed from the emails before performing clustering process. Suffix-stripping algorithm is also used to perform stemming. The data sets' details are shown in table 1.

4.3 Computational Time and Quality of Clustering

We compare our clustering results with other clustering algorithms. The numbers of main clusters produced by each algorithm are shown in table 2. Here we choose the clusters with email number greater than 5 as main clusters because grouping emails into even smaller clusters will require people's cognitive load to search through a large number of groups and therefore those clusters become useless. From this table we can see our algorithm can match the labelled data set 5 and 6 very well in terms of cluster number.

Table 1. Details of data set

Data Set No.	Labelled by human	Number of emails	Number of clusters	Number of words
1	No	1023	-	17650
2	No	1235	-	16432
3	No	2045	-	30215
4	No	3987	-	33442
5	Yes	342	7	2032
6	Yes	1126	12	7839

Table 2. Numbers of main clusters

Data Set	Nonparametric Text Clustering	Hierarchical agglomerative	K-means
1	14	16	10
2	13	15	10
3	15	17	12
4	16	15	15
5	7	12	7
6	12	13	12

Table 3. Computational time (seconds)

Data Set	Nonparametric Text Clustering	Hierarchical agglomerative	K-means
1	28.7	232	98.2
2	20.3	212	97.5
3	58.7	538	211
4	135	1207	484
5	9.21	101	42.1
6	22.2	215	103

Table 4. Hubert's Γ statistic

Data Set	Nonparametric Text Clustering	Hierarchical agglomerative	K-means
1	0.764	0.563	0.329
2	0.793	0.542	0.321
3	0.821	0.598	0.332
4	0.866	0.457	0.319
5	0.902	0.788	0.438
6	0.791	0.554	0.337

Table 3 shows the computational time of each algorithm. The time unit is second. From the table we can see our nonparametric text clustering algorithm performs much faster than both hierarchical agglomerative algorithm and k-means algorithm. For example, for data set 1, hierarchical agglomerative algorithm needs 808% of time of our algorithm to perform the clustering, and k-means algorithm needs 342% of time of our algorithm to perform the clustering. From the absolute computational time point of view, our algorithm also costs reasonably low. For example, when classifying data set 4 with 3987 emails, the computation time is 135 seconds (about 2 minutes), which is a reasonable response for a clustering system. We also find that k-means algorithm is faster than hierarchical agglomerative algorithm in all the runs, which conforms that k-means has lower time complexity than hierarchical agglomerative algorithm.

The average Hubert's Γ statistic of each algorithm is shown in table 4. From the table we can see for all the data sets, the Hubert's Γ statistic is higher than 0.764 if our clustering algorithm is used. Our algorithm outperforms others in this measurement, which means the clusters produced by our algorithm have higher intra-cluster similarity and lower inter-cluster similarity than other algorithms. Hierarchical agglomerative algorithm is better than k-means algorithm in terms of clustering quality measured by Hubert's Γ statistic.

The average simple matching coefficient of each algorithm is shown in table 5. Again we find our algorithm has better clustering quality measured by simple matching coefficient than both hierarchical agglomerative algorithm and k-means algorithm. The simple matching coefficient of k-means is lower than both our algorithm and hierarchical agglomerative algorithm.

The average Jaccard coefficient of each algorithm is shown in table 6. The Jaccard coefficient is often more sensitive than the simple matching coefficient sometimes the

negative matches are a dominant factor. From the table we find our algorithm achieved above 0.821 Jaccard coefficients for all the data sets. Hierarchical agglomerative algorithm has slightly lower Jaccard coefficient than our algorithm from data set 1, 4, and 5. It has slightly higher Jaccard coefficient than our algorithm from data set 2, 3, and 6. K-means has much lower Jaccard coefficient than both our algorithm and hierarchical agglomerative algorithm.

From the above results we can clearly see our text clustering algorithm outperforms traditional hierarchical agglomerative algorithm and k-means algorithm in terms of computational time and clustering quality which is measured by Hubert's Γ statistic, simple matching coefficient, and Jaccard coefficient.

Table 5. Simple matching coefficient

Data Set	Nonparametric Text Clustering	Hierarchical agglomerative	K-means
1	0.978	0.912	0.839
2	0.974	0.923	0.856
3	0.964	0.918	0.832
4	0.939	0.921	0.813
5	0.991	0.985	0.903
6	0.971	0.922	0.847

Table 6. Jaccard coefficient

Data Set	Nonparametric Text Clustering	Hierarchical agglomerative	K-means
1	0.883	0.821	0.475
2	0.821	0.832	0.442
3	0.842	0.846	0.398
4	0.881	0.826	0.338
5	0.899	0.853	0.491
6	0.823	0.827	0.448

5 Related Work

Related work on techniques for managing email overload can be found in [4, 6, 12, 13]. We find that all existing email management systems heavily rellies on a user-created folder structure or user-defined input parameters, which have not essentially achieved the goal of automatically managing email overload.

6 Conclusion

Email overload problem has strongly affect people's usage of email as a knowledge management tool. We proposed a novel email clustering system to solve this problem. This system is essentially supported by a new automatic nonparametric clustering algorithm. By using this algorithm, emails users can get clustered emails easily

without any input. The experiments show our algorithm has high efficiency and high clustering quality in terms of computation time and clustering quality measured by Hubert's Γ statistic, simple matching coefficient, and Jaccard coefficient.

References

1. IDC: IDC Examines the Future of Email As It Navigates Security Threats, Compliance Requirements, and Market Alternatives (2005), http://www.idc.com/getdoc.jsp?containerId=prUS20033705
2. Schultze, U., Vandenbosch, B.: Information Overload in a Groupware Environment: Now You See It, Now You Don't. Journal of Organizational Computing and Electronic Commerce 8, 127–148 (1998)
3. Schuff, D., Turetken, O., D'Arcy, J., Croson, D.: Managing E-Mail Overload: Solutions and Future Challenges. IEEE Computer 40, 31–36 (2007)
4. Schuff, D., Turetken, O., D'Arcy, J.: A Multi-attribute, Multi-weight Clustering Approach to Managing, E-Mail Overload. Decision Support Systems 42, 1350–1365 (2006)
5. Roussinov, D.G., Chen, H.: Document Clustering for Electronic Meetings: An Experimental Comparison of Two Techniques. Decision Support Systems 27, 67–79 (1999)
6. Mock, K.: An Experimental Framework for Email Categorization and Management. In: 24th ACM International Conference on Research and Development in Information Retrieval, pp. 392–393 (2001)
7. Whittaker, S., Sidner, C.: Email Overload: Exploring Personal Information Management of Email. In: ACM SIGCHI conference on Human Factors in Computing Systems, pp. 276–283 (1996)
8. Baker, F.B., Hubert, L.J.: Measuring the Power of Hierarchical Cluster Analysis. Journal of the American Statistical Association 70, 31–38 (1975)
9. Aldenderfer, M.S., Blashfield, R.K.: Cluster Analysis. Sage Publications (1984)
10. Tabachnick, B.G., Fidell, L.S.: Using Multivariate Statistics. Harper Collins College Publishers, New York (1996)
11. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley, Reading (1999)
12. Payne, T., Edwards, P.: Interface Agents that Learn: An Investigation of Learning Issues in a Mail Interface. Applied Artificial Intelligence 11, 1–32 (1997)
13. Kushmerick, N., Lau, T.: Automated E-Mail Activity Management: An Unsupervised Learning Approach. In: 10th International Conf. on Intelligent User Interfaces, pp. 67–74 (2005)