Vol. 3, No. 2
April - June 2007

**An official publication of
the Information Resources
Management Association**

**Special Issue:
Web Mining**

INTERNATIONAL JOURNAL OF INTELLIGENT INFORMATION TECHNOLOGIES

IDEA GROUP PUBLISHING

Publisher of IT books, journals and cases since 1988
www.idea-group.com

IGI PUBLISHING
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.igi-pub.com

ITJ3628

This paper appears in the publication, **International Journal of Intelligent Information Technologies**, Volume 3, Issue 2
edited by **Vijayan Sugamaran © 2007, IGI Global**

Clustering Web Pages into Hierarchical Categories

Zhongmei Yao, Louisiana Tech University, USA

Ben Choi, Louisiana Tech University, USA

ABSTRACT

Clustering is well suited for Web mining by automatically organizing Web pages into categories each of which contains Web pages having similar contents. However, one problem in clustering is the lack of general methods to automatically determine the number of categories or clusters. For the Web domain, until now there is no such a method suitable for Web page clustering. To address this problem, we discovered a constant factor that characterizes the Web domain, based on which we propose a new method for automatically determining the number of clusters in Web page datasets. We also propose a new Bidirectional Hierarchical Clustering algorithm, which arranges individual Web pages into clusters and then arranges the clusters into larger clusters and so on until the average inter-cluster similarity approaches the constant factor. Having the new constant factor together with the new algorithm, we have developed a clustering system suitable for mining the Web.

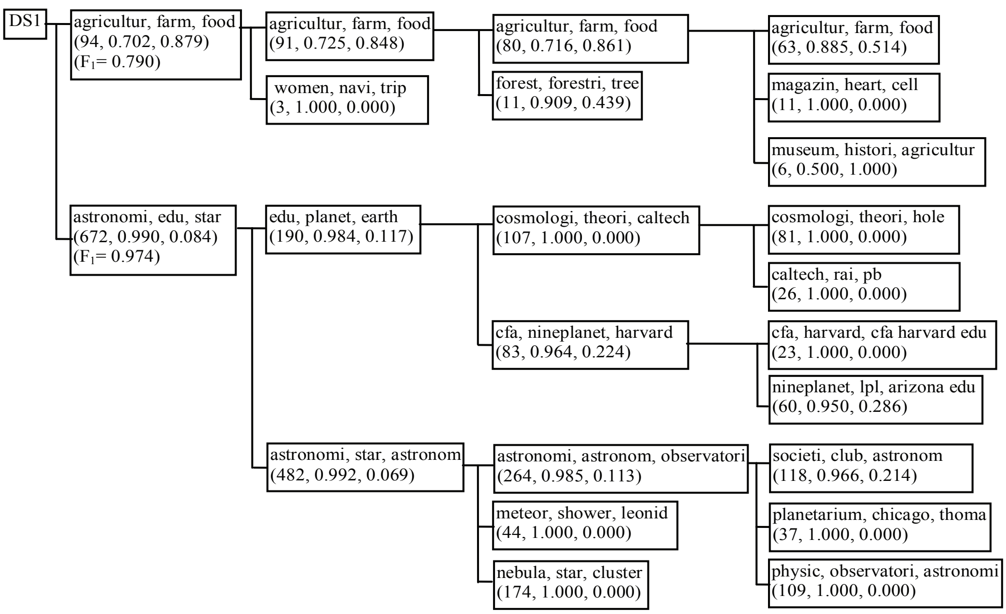
Keywords: information retrieval; knowledge classification; knowledge discovery; Semantic Web; Web mining

INTRODUCTION

We are interested in cluster analysis that can be used to organize Web pages into clusters based on their contents or genres (Choi & Yao, 2005). Clustering is an unsupervised discovery process for partitioning a set of data into clusters such that data in the same cluster is more similar to one another than data in other clusters (Berkhin, 2002; Everitt et al., 2001; Jain & Dubes, 1998; Jain et al., 1999). Typical application areas for clustering include artificial intelligence, biology, data mining, information retrieval, image processing, marketing, pattern recognition, and statistics (Berkhin, 2002; Everitt et al., 2001; Jain et al., 1999). Compared to classification methods, cluster analysis has the advantage that it does not require any training data (i.e., the labeled data), but can achieve the same goal in that it can classify similar Web pages into groups.

The major aspects of the clustering problem for organizing Web pages are: To find the number of clusters *k* in a Web page dataset, and to assign Web pages accurately to their clusters. Much work (Agrawal et al., 1998; Dhillon et al., 2001; Ester et al., 1996; Guha et al., 1998a; Guha et al., 1998b; Hinneburg & Keim, 1999; Karypis & Kumar, 1999; Ng & Han, 1994; Rajaraman & Pan, 2000; Sander et al., 1998; Tantrum et al., 2002; Yao & Karypis, 2001; Zhang et al., 1996; Zhao & Karypis, 1999) has been done to improve the accuracy of assigning data to clusters in different domains, whereas no satisfactory method has been found to estimate *k* in a dataset (Dudoit & Fridlyand, 2002; Strehl, 2002) though many methods were proposed (Davies & Bouldin, 1979; Dudoit & Fridland, 2002; Milligan & Cooper, 1985). As a matter of fact, finding *k* in a dataset is still a challenge in cluster analysis (Strehl, 2002). Almost all existing work in this area assumes that *k* is known for clustering a dataset (e.g., Karypis et al., 1999; Zhao & Karypis, 1999). However in

Figure 1. The hierarchical structure produced for dataset DS1. Each box in this figure represents a cluster. The format of the description of a cluster is: its top three descriptive terms followed by (#docs, purity, entropy). Only the descriptions of clusters at the top level contain the F_1 scores.



many applications, this is not true because there is little prior knowledge available for cluster analysis except the feature space or the similarity space of a dataset.

This article addresses the problem of estimating *k* for Web page datasets. By testing many existing methods for estimating *k* for datasets, we find that only the average inter-cluster similarity (*avgInter*) need to be used as the criterion to discover *k* for a Web page dataset. Our experiments show that when the *avgInter* for a Web page dataset reaches a constant threshold, the clustering solutions for different datasets from the Yahoo! directory are measured to be the best. Compared to other criterion, e.g., the maximal or minimal inter-cluster similarity among clusters, *avgInter* implies a characteristic for Web page datasets.

This article also describes our new clustering algorithm called bi-directional hierarchical clustering. The new clustering algorithm arranges individual Web pages into clusters and then arranges the clusters into larger clusters and so on until the average inter-cluster similarity approaches a constant threshold. It produces a hierarchy of categories (see for example Figure 1), in which larger and more general categories locate at the top while smaller and more specific categories locate at the bottom of the hierarchy. Figure 1 shows the result of one of our experiments for clustering 766 Web pages to produce a hierarchy of categories. The top (left-most) category contains all the Web pages (Dataset 1). The next level consists of two categories, one of which has 94 Web pages and the other has 672 Web pages. Then, each of the two categories has sub-categories and so on, as shown in the figure. This example shows that our new clustering algorithm is able to handle categories of widely different sizes (such as 94 comparing to 672 pages). By using two measures, purity and entropy, this example also shows that more general categories (which have lower purity but higher entropy) locate at the top, while more specific categories (which have higher purity but lower entropy) locate at the bottom of the hierarchy.

The rest of this article is organized as follows. The second section gives background and an overview of related methods. Our new bi-directional hierarchical clustering algorithm is presented in the third section. The fourth section describes the Web page datasets used in our experiments. The fifth section provides the experimental details for the discovery of a constant factor that characterizes the Web domain. The sixth section shows how the constant factor is used for automatically discovering the number of clusters. The seventh section provides the conclusion and future research.

BACKGROUND AND RELATED METHODS

In this section we first give the necessary background of cluster analysis and then briefly review existing methods for estimating the number of clusters in a dataset.

The task of clustering can be expressed as follows (Berkhin, 2002; Everitt et al., 2001; Jain et al., 1999). Let n be the number of objects, data points, or samples in a dataset, m the number of features for each data point d_i with $i \in \{1,...,n\}$, and k be the desired number of clusters to be recovered. Let $l \in \{1,...,k\}$ denote the unknown cluster label and C_l be the set of all data points in the l cluster. Given a set of m-dimensional data points, the goal is to estimate the number of clusters k and to estimate the cluster label l of each data point such that similar data points have the same label. Hard clustering assigns a label to each data point while soft clustering assigns the probabilities of being a member of each cluster to each data point. In the following we present an overview of several common methods for estimating k for a dataset.

Calinski and Harabasz (1974) defined an index, *CH(k)*, to be:

$$CH(k) = \frac{trB(k)/(k-1)}{trW(k)/(n-k)} \quad (1)$$

where *tr* represents the trace of a matrix, *B(k)* is the between cluster sum of squares with k clusters and *W(k)* is the within cluster sum of squares with k clusters (Mardia et al., 1979). The number of clusters for a dataset is given by arg $\max_{k\geq 2} CH(k)$.

Krzanowski and Lai (1985) defined the following indices for estimating k for a dataset:

$$diff(k) = (k-1)^{2/m} trW_{k-1} - k^{2/m} trW_k \quad (2)$$

$$KL(k) = \frac{|diff(k)|}{|diff(k+1)|} \quad (3)$$

where m is number of features for each data point. The number of clusters for a dataset is estimated to be arg $\max_{k\geq 2} KL(k)$.

The Silhouette width is defined (Kaufman & Rousseeuw, 1990) to be a criterion for estimating k in a dataset as follows:

$$sil(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \quad (4)$$

where *sil(i)* means the Silhouette width of data point i, *a(i)* denotes the average distance between i and all other data in the cluster which i belongs to, and *b(i)* represents the *smallest* average distance between i and all data points in a cluster. The data with large *sil(i)* is well clustered. The overall average silhouette width is defined by $\overline{sil} = \sum_i sil_i / n$ (where n is the number of data in a dataset). Each k ($k \geq 2$) is associated with a $\overline{sil_k}$ and the k is selected to be the right number of clusters for a dataset which has the largest $\overline{sil}$ (i.e. $k = \arg\max_{k\geq 2} \overline{sil_k}$).

Similarly, Strehl (2002) defined the following indices:

$$avgInter(k) = \sum_{i=1}^{k} \frac{n_i}{n - n_i} \sum_{j \in \{1,...,i-1,i+1,...,k\}} n_j \cdot Inter(C_i, C_j) \quad (5)$$

$$avgIntra(k) = \sum_{i=1}^{k} n_i Intra(C_i) \quad (6)$$

$$\varphi(k) = 1 - \frac{avgInter(k)}{avgIntra(k)} \quad (7)$$

where *avgInter(k)* denotes the weighted average inter-cluster similarity, *avgIntra(k)* denotes the weighted average intra-cluster similarity, $Inter(C_i, C_j)$ means the inter-cluster similarity between cluster C_i with n_i data points and cluster C_j with n_j data points, $Intra(C_i)$ means the intra-cluster similarity within cluster C_i, and $\phi(k)$ is the criterion designed to measure the quality of clustering solution. The $Inter(C_i, C_j)$ and $Intra(C_i)$ are given by (Strehl, 2002)

$$Inter\,(C_i, C_j) = \frac{1}{n_i n_j} \sum_{d_a \in C_i, d_b \in C_j} sim(d_a, d_b) \quad (8)$$

$$Intra\,(C_i) = \frac{2}{(n_i - 1)n_i} \sum_{d_a, d_b \in C_i} sim(d_a, d_b) \quad (9)$$

where d_a and d_b represent data points. To obtain high quality with small number of clusters, Strehl (2002) also designed a penalized quality $\phi^T(k)$ which is defined as

$$\varphi^T(k) = (1 - \frac{2k}{n})\varphi(k). \quad (10)$$

The number of clusters in a dataset is estimated to be arg $\max_{k \geq 2} \phi^T(k)$.

It can be noticed that the above methods cannot be used for estimating k=1 for a dataset. Some other methods, e.g., Clest (Dudoit & Fridlyand, 2002), Hartigan (1985), and gap (Tibshirani et al., 2000) were also found in literature.

In summary, most existing methods make use of the distance (or similarity) of inter-cluster and (or) intra-cluster of a dataset. The problem is that none of them is satisfactory for all kinds of cluster analysis (Dudoit & Fridlyand, 2002; Stehl, 2002). One reason may be that people have different opinions about the granularity of clusters and there may be several right answers to *k* with respect to different desired granularity. Unlike partitional (flat) clustering algorithms, hierarchical clustering algorithms may have different *k*'s by cutting the dendrogram at different levels, hence providing flexibility for clustering results.

In the next section we will present our new clustering algorithm which is used to cluster Web pages and to estimate *k* for Web page datasets. Throughout this article, we use term "documents" or "Web pages" to denote Web pages, the term "true class" to mean a class of Web pages which contains Web pages labeled with the same class label, and the term "cluster" to denote a group of Web pages in which Web pages may have different class labels.

Bi-Directional Hierarchical Clustering Algorithm

We present our new bi-directional hierarchical clustering (BHC) system (Yao & Choi, 2003) in this section. The BHC system consists of three major steps:

1. Generating an initial sparse graph,

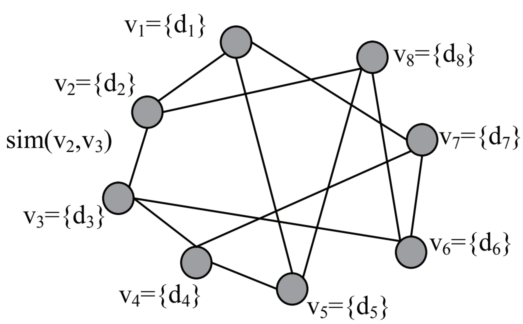
2. Bottom-up cluster-merging phase, and
3. Top-down refining phase.

These major steps are described in detail in the following subsections. Here we outline the workings of the entire system. In the first phase, the BHC system takes a given dataset and generates an initial sparse graph (e.g., Figure 2), where a node represents a cluster, and is connected to its k-nearest neighbors by similarity-weighted edges. The BHC system then creates a hierarchical structure of clusters in the two phases, the bottom-up cluster-merging phase and the top-down refinement phase. During the bottom-up cluster-merging phase, two or more nodes (clusters) are merged together to form a larger cluster. Then, again two or more clusters are merged and so on until a stopping condition is met. During the top-down refinement phase, the BHC system eliminates the early errors that may occur in the greedy bottom-up cluster-merging phase. It moves some nodes between clusters to minimize the inter-cluster similarities. Thus, these two phases make items in a cluster more similar and make clusters more distinct from each other. The key features of the BHC system are that it produces a hierarchical structure of clusters much faster than the existing hierarchical clustering algorithms, and it improves clustering results using a refinement process, as detailed in the following.

Generating an Initial Sparse Graph

In this subsection we describe how to arrange a set of Web pages to form a weighted graph (e.g., Figure 2) based on the similarities of Web pages. A Web page is first converted to a vector of terms:

Figure 2. The initial all-k-nearest-neighbor (Aknn) graph G_0 with n nodes (n=8 in this case). Each node in this graph contains a single Web page (e.g., node v_1 contains Web page d_1) and is connected to its k-nearest neighbors (k is 3 in this case). The edge connecting two nodes is weighted by the similarity between the two nodes.



$$d_i = (w_{i1}, \ldots, w_{ij}, \ldots, w_{im}) \quad (11)$$

where Web page d_i has m terms (also called features), and the weights of the features are indexed from w_{i1} to w_{im}. Usually a feature consists of one to three words, and its weight is the number of occurrences of the feature in a Web page. Common methods to determine w_{ij} are the term frequency-inverse document frequency (tf-idf) method (Salton & Buckley, 1988) or the structure-oriented term weighting method (Peng, 2002; Riboni, 2002). Many approaches (e.g., Rijsbergen, 1979; Strehl et al., 2000) are then used to measure the similarity between two Web pages by comparing their vectors. We choose the cosine (Rijsbergen, 1979) as the metric for measuring the similarity, i.e., $cos(d_i, d_j)$ is the cosine similarity between Web pages d_i and d_j. We then define the similarity between two clusters u and v as:

$$sim(u,v) = \frac{\sum_{d_i \in u, d_j \in v} \cos(d_i, d_j)}{|u||v|} \quad (12)$$

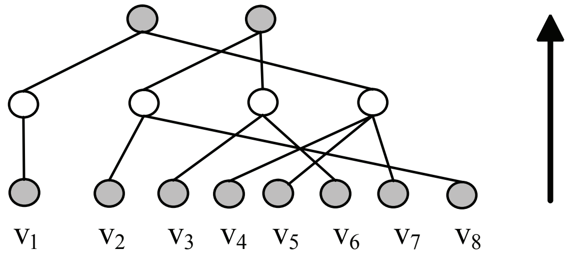
where d_i is a Web page within cluster u, d_j is a Web page within cluster v, $|u|$ is the number of Web pages in cluster u.

An initial sparse graph is generated by using the *sim(u, v)* to weight the edge between two nodes u and v. Figure 2 shows an example. Initially each node in the graph contains only one Web page. Each node does not connected to all other nodes, but only to k most similar nodes. By choosing k small in comparison to the total number of nodes, we can reduce the computation time in later clustering processes.

Bottom-Up Cluster-Merging Phase

In the bottom-up cluster-merging phase we aim at maximizing the intra-similarities of clusters by merging the most similar clusters together (see Figure 3 for example). To achieve this goal,

Figure 3. Illustration of the bottom-up cluster-merging procedure. The nodes at the same level are nodes in a same graph. Some nodes at the lower level are merged to form a single node at the higher level. The two nodes at the top level represent the two final clusters in this example.



we transform the initial sparse graph G_0 into a sequence of smaller graphs G_1, G_2, …, G_t such that the number of nodes $|V_0|>|V_1|>|V_2|> \ldots >|V_t|$, where a stopping criteria is met at G_t. The nodes in the smallest graph G_t represent the final clusters for a dataset.

We first define the most similar neighborhood of a node v, $N_v(\delta_i)$, to be a set of nodes fulfilling the following condition:

$$N_v(\delta_i) = \{u \mid sim(v,u) > \delta_i \} \quad (13)$$

where $sim(v, u)$ is the similarity between node v and node u (see Equation 12), and δ_i is an adaptive threshold (e.g., δ_i =0.543) and is associated with graph G_i. The nodes within $N_v(\delta_i)$ of node v in G_i are merged together to become a new node in the smaller graph G_{i+1} (illustrated in Figure 3). The number of nodes and the number of edges in the smaller graph are reduced, and the number of Web pages in a node in the smaller graph G_{i+1} is increased, resulting in grouping similar Web pages into nodes (or clusters).

After new nodes in the smaller graph G_{i+1} are formed, the edges between nodes are built under two conditions: (1) similarity between two nodes is greater than zero and (2) a new node is connected to at most k most similar nodes. Furthermore, since $N_v(\delta_i) \subseteq Nv(\delta_{i+1})$ whenever $\delta_i \geq \delta_{i+1}$, we design

$$\delta_{i+1} = \delta_i / \beta \quad (14)$$

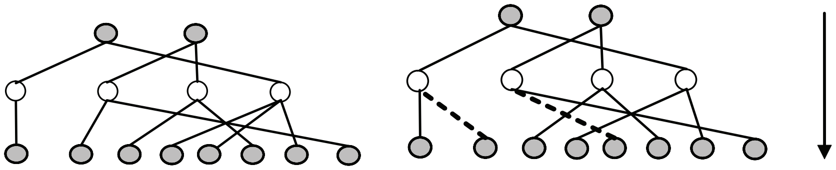
where $\beta>1$ is a decay factor (Rajaraman & Pan, 2000), which defines a weaker neighborhood for the smaller graph G_{i+1} in order to continue to transfer G_{i+1} into another smaller graph. Therefore this is an iterative procedure to transfer the initial graph G_0 to the sequence of smaller graph G_1, G_2, …, G_t such that $|V_0|>|V_1|>|V_2|> \ldots >|V_t|$. The decay factor β controls the speed of reducing the value of threshold δ in a way that $\delta_0 = 1/\beta$, $\delta_1 = \delta_0/\beta$, …, $\delta_t = \delta_{t-1}/\beta$. The faster the value of δ is reduced, the more nodes in the current graph G_i may be grouped to be a new node in the next smaller graph G_{i+1}, producing less new nodes in G_{i+1}. Therefore the decay factor β determines the speed of reducing the number of the sequence of smaller graphs. A larger β will result in a fewer number of levels in the hierarchical structure.

A stopping factor is required to terminate this bottom-up cluster-merging procedure. The details for the discovery of a stopping factor for Web page datasets are provided in the fifth section. This bottom-up cluster-merging phase is a greedy procedure, which may contain errors or fall into local minima. To address this problem, we apply a top-down refinement procedure.

Top-Down Refinement Phase

The top-down refinement phase refines the greedy clustering results produced by the bottom-up cluster-merging phase (see Figure 4 for example). The objective in this phase is to make clusters more distinct from each other.

Figure 4. Illustration of top-down refinement procedure. (a) Shows the bottom-up clustering solution, which is used to compare the improvement produced by the top-down refinement procedure. (b) Shows the final clustering solution after the top-down refinement procedure. The dashed lines in (b) indicate the error correction. The hierarchical structure in (b) can be used for users to browse Web pages.



We first define the term sub-node: a sub-node *s* of a node *u* in a graph G_i+1 is a node *s* in graph G_i. For instance in Figure 5, node *x* is a sub-node of node *u*. The top-down refinement procedure operates on the following rule: If a sub-node *x* of a node *u* is moved into another node *v* and this movement results in reduction of the inter-similarity between the two nodes, then the sub-node *x* should be moved into the node *v*. The reduction of the inter-similarity between two nodes, *u* and *v*, by moving a sub-node *x* from node *u* to node *v* can be expressed by a gain function which is defined as:

$$gain_x(u,v) = sim(u,v) - sim((u-x),(v+x)) \quad (15)$$

where *u-x* means the node after removing sub-node *x* out of *u*, and *v+x* means the node after adding sub-node *x* into *v*. Although a sub-node is considered to be moved into any of its connected nodes, it is moved only to its connected node that results in the greatest positive gain. To keep track of the gains, a gain list is used and its implementation can be found in, e.g., Fiduccia and Mattheyses (1982).

Our refinement procedure refines clustering solution from the smallest graph, G_t, at the top level to the initial graph, G_0, at the lowest level (see Figure 4). Sub-nodes are moved until no more positive gain will is obtained. For the example shown in Figure 4, two sub-nodes are moved to different clusters.

This refinement procedure is very effective in climbing out of local minima (Hendrickson & Leland, 1993; Karypis & Kumar, 1999). It not only finds early errors produced by the greedy cluster-merging procedure, but also can move groups of Web pages of different sizes from one cluster into another cluster so that the inter-cluster similarity is reduced.

The nodes in graph G_t at the top level in the hierarchical structure (see

Figure 5. Moving a sub-node x into its connected node with the greatest gain

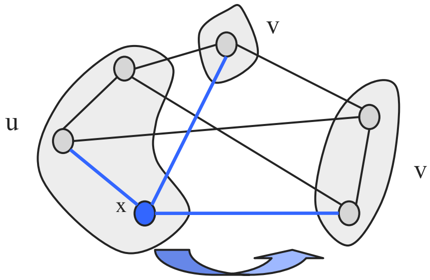


Figure 4) generated after the top-down refinement procedure represent final clusters for a dataset. The resultant hierarchical structure can be used for Web browsing, with larger and more general clusters at higher levels while smaller and more specific clusters are at lower levels.

Web Page Datasets for Experiments

For testing our bi-directional hierarchical clustering algorithm and for discovering a new constant stopping factor, we conducted a number of experiments on Web page datasets. Here we report four Web page datasets taken from Yahoo.com (see Table 1) representing datasets with different sizes and different granularity and we skip other datasets for brevity since their experimental results were found to have similar quality. The first dataset, *DS1*, contains 766 Web pages which are randomly selected from two true classes: *agriculture* and *astronomy*. This dataset is designed to show our method of estimating the number of clusters *k* in a dataset which consists of clusters of widely different sizes: The number of Web pages from the *astronomy* true class is about ten times the number of Web pages from the *agriculture* true class. The second dataset, *DS2*, contains 664 Web pages from 4 true classes. The third dataset, *DS3*, includes 1215 Web pages from 12 true classes. In order to show the performance on a more diverse dataset, we produce the forth dataset, *DS4*, which consists of 2524 Web pages from 24 true classes. After we remove stop words and conduct reduction of dimensionality (Yao, 2004), the final dimension for each dataset is listed in Table 1.

Discovery of a Constant Factor

In this section, we outline our experiments for the discovery of a constant factor that characterizes the Web domain and makes our clustering algorithm applicable for clustering Web pages. For all experiments, we use the metric, F_1 measure (Larsen & Aone, 1999; Zhao & Karypis, 1999), which makes use of true class labels of Web

Table 1. Compositions of four representative Web page datasets

DS1: true classes = 2, the number of web pages= 766, dimension= 1327

true class (the number of web pages):
agriculture(73) astronomy(693)

DS2: true classes = 4, the number of web pages=664, dimension=1362

astronomy(169) biology(234) alternative(119) mathematics(142)

DS3: true classes = 12, the number of web pages = 1215, dimension= 1543

agriculture(108) astronomy(92) evolution(74) genetics(108) health(127) music(103) taxes(80) religion(113) sociology(110) jewelry(108) network (101) sports(91)

DS4: true classes = 24, the number of web pages = 2524, dimension= 2699

agriculture(87) astronomy(96) anatomy(85) evolution(76) plants(124) genetics(106) mathematics(106) health(128) hardware(127) forestry(68) radio(115) music(104) automotive(109) taxes(82) government(147) religion(114) education(124) art(101) sociology(108) archaeology(105) jewelry(106) banking(72) network (88) sports(146)

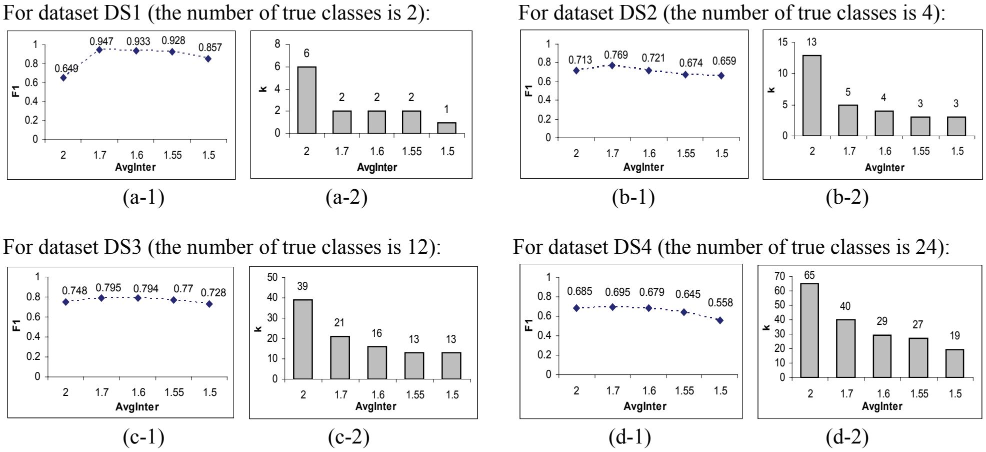
pages, to measure the quality of clusters in a Web page dataset. The F_1 measure indicates how well a cluster solution matches the true classes in the real world (e.g., the Yahoo! directory). In general, the greater F_1 score, the better clustering solution.

In our experiments we test the existing methods *CH(k)*, *KL(k)*, $\overline{sil_k}$, $\phi(k)$ and $\phi^T(k)$ (see the second section) to discover the number of clusters *k* for Web page datasets. These five metrics are computed for different *k*'s for a Web page dataset. However, none of them works well. Our tests results showed that for any dataset in Table 1 their estimated *k* is more than 5 times different from the true number of classes in the Web page datasets and the corresponding cluster solutions have a lower than 0.3 F_1 score.

After many trials, we find that *avgInter(k)* for any dataset in Table 1 reaches a common threshold of 1.7, when the F_1 measure of the cluster solution for a dataset is greatest. The relation between the thresholds of *avgInter(k)* and the F_1 scores of a cluster solution, and the relation between the thresholds of *avgInter(k)* and *k*'s for the four Web page datasets are illustrated in Figure 6.

In Figure 6 (a-1), (b-1), (c-1) and (d-1), the F_1 scores of cluster performances for the four datasets reach the maximal values when the threshold of *avgInter* is 1.7, and further increasing or reducing the threshold of *avgInter*

Figure 6. The impact of avgInter on clustering performances for four representative Web page datasets



would only worsen the F_1 scores for the datasets *DS1*, *DS2*, *DS3* and *DS4*. In other words, once the weighted average inter-cluster similarity (*avgInter*) reaches the common threshold, 1.7, the cluster solution is found to be best for a Web page dataset. This shows that, unlike other metric such as *CH(k)*, *KL(k)*, $\overline{sil}_k$, or $\phi^T(k)$, *avgInter* implies a common characteristic in different Web page datasets.

Figure 6 (a-2), (b-2), (c-2) and (d-2) show the *k*'s for four Web page datasets produced by setting different thresholds for *avgInter*. In Figure 6 (a-2) it is shown that the *avgInter* method is able to find *k*=1 while many existing methods are unable to do so. As shown in the figure, when *avgInter* reaches 1.7, the best estimated values for *k* is found to be 2 for *DS1*, 5 for *DS2*, 21 for *DS3* and 40 for *DS4*.

The estimated *k* is usually greater than the number of true classes in a Web page dataset because outliers are found and clustered into some small clusters, and a few true classes are partitioned into more than one cluster with finer granularity. This situation is exactly shown in Table 2, which shows the clustering solution for the most diverse dataset, *DS4*, obtained when the threshold of *avgInter* is 1.7. The naming for a newly formed cluster is by selecting the top three descriptive terms. The ranking of descriptive terms for a cluster is conducted by sorting the tf_{lj}'/df_j values of terms in the cluster (tf_{lj}' is defined to be the number of Web pages containing term t_j in cluster C_l and df_j is the document frequency (Yang & Pedersen, 1997) of t_j). It can be noted that for most true classes, a true class has a dominant cluster in Table 2. For

Table 2. The clustering solution for dataset DS4.

cluster	The number of web pages	the majority's true class label	purity	F_1	top 3 descriptive terms
C_1	106	Astronomy	0.840	0.881	moon, mar, orbit
C_2	29	Agriculture	0.793	0.397	pest, weed, pesticid
C_3	24	Agriculture	0.917		crop, wheat, agronomi
C_4	64	Anatomy	0.906	0.779	anatomi, muscl, blood
C_5	64	Evolution	0.750	0.686	evolut, darwin, erectu
C_6	116	Plants	0.776	0.750	plant, flower, garden
C_7	161	Genetics	0.565	0.682	genom, genet, clone
C_8	101	Mathematics	0.782	0.763	mathemat, math, algebra
C_9	94	Health	0.649	0.550	mental, therapi, health
C_{10}	32	Health	0.875		grief, bereav, heal
C_{11}	115	Hardware	0.452	0.430	font, px, motherboard
C_{12}	21	Hardware	0.857		keyboard, pc, user
C_{13}	83	Forestry	0.675	0.742	forest, forestri, tree
C_{14}	86	Radio	0.709	0.607	radio, broadcast, fm
C_{15}	70	Music	0.800	0.644	guitar, music, instrum
C_{16}	13	Music	1.000		drum, rhythm, indian
C_{17}	86	Automotive	0.849	0.749	car, auto, automot
C_{18}	20	Automotive	0.800		motorcycl, bike, palm
C_{19}	120	Taxes	0.633	0.752	tax, incom, revenu
C_{20}	155	Government	0.806	0.828	congressman, hous, district
C_{21}	108	Religion	0.824	0.802	christian, bibl, church
C_{22}	92	Education	0.761	0.648	montessori, school, educ
C_{23}	43	Education	0.767		homeschool, home school, curriculum
C_{24}	60	Art	0.833	0.621	paint, canva, artist
C_{25}	89	Sociology	0.831	0.751	sociologi, social, sociolog
C_{26}	59	Archaeology	0.864	0.622	archaeologi, archaeolog, excav
C_{27}	18	Archaeology	0.722		egypt, egyptian, tomb
C_{28}	120	Jewelry	0.817	0.867	jewelri, bead, necklac
C_{29}	91	Banking	0.659	0.736	bank, banker, central bank
C_{30}	92	Network	0.565	0.578	network, dsl, storag

(F_1 scores are given only for 24 clusters because those clusters represent true classes in dataset DS4. The purity (Strehl et al., 2000) and the top three descriptive terms are given for each cluster.)

Table 2. cotinued

C_{31}	159	Sports	0.824	0.859	soccer, footbal, leagu
C_{32}	1	Religion	1.000		struggl, sex, topic
C_{33}	8	Religion	0.250		domain, registr, regist
C_{34}	10	Plants	0.300		florida, loui, ga, part, pioneer,
C_{35}	1	Archaeology	1.000		guestbook, summari, screen
C_{36}	3	Genetics	0.333		pub, patch, demo
C_{37}	3	Music	0.333		bell, slide, serial
C_{38}	1	Sociology	1.000		relief, portrait, davi
C_{39}	2	Music	0.500		ontario, predict, archaeolog
C_{40}	4	Music	0.250		unix, php, headlin
overall	2524		0.740	0.698	

(F_1 scores are given only for 24 clusters because those clusters represent true classes in dataset DS4. The purity (Strehl et al., 2000) and the top three descriptive terms are given for each cluster.)

instance, the dominant clusters for true class *astronomy, anatomy* and *evolution* are cluster C_1, C_4 and C_5, respectively. We can see several true classes have been partitioned more precisely into more than one cluster; e.g., true class *automotive* has been separated into cluster C_{17} which is more related to *car* and *auto*, and cluster C_{18} more related *motorcycle* and *bike*, as indicated by their top descriptive terms. Similar situation happens to true class *agriculture*, *health*, *education* and *archaeology*, each of which has been partitioned into two clusters. As shown in Table 2, outliers, which have low purity scores, can be found as cluster C_{32}, C_{33}, …, and C_{40}.

Discovering the Number of Clusters

The constant factor described in the last section can be used to estimate the number of clusters in a clustering process. The number of clusters k for a Web page dataset is estimated to be:

$$\arg\max_k (avgInter(k) \leq 1.7)$$

where $1 \leq k \leq n$. (16)

The $avgInter(k)$ is computed for different k's. The k that results in $avgInter(k)$ as close to (but less than) the threshold 1.7 is selected to be the final k for a Web page dataset.

For our bi-directional hierarchical clustering system, we determine the number of clusters by using the constant as the stopping factor in the clustering process. Our hierarchical clustering process starts by arranging individual Web pages into clusters and then arranging the clusters into larger clusters and so on until the average inter-cluster similarity *avgInter*(k) approaches the constant. As clusters are grouped to form larger clusters the value of *avgInter*(k) is reduced. This grouping process (bottom-up cluster-merging phase) is stopped when *avgInter*(k) approaches 1.7. The final number of clusters is automatically obtained as the result.

CONCLUSION AND FUTURE RESEARCH

Although many methods of finding the number of clusters for a dataset have been proposed, none of them is satisfactory for clustering Web page datasets. Finding the number of clusters for a dataset is often treated as an ill-defined question because it is still questionable how well a cluster should be defined. By recognizing this status, we preferred hierarchical clustering methods, which allow us to view clusters at different levels with coarser granularity at the higher level and finer granularity at the lower level. For Web mining in particular, our Bidirectional Hierarchical Clustering method is able to arrange Web pages into a hierarchy of categories that allows users to browse the results in different levels of granularities.

Besides proposing the new Bi-directional Hierarchical Clustering algorithm, we investigated the problem of estimating the number of clusters, k, for Web page datasets. After many trials, we discovered that the average inter-cluster similarity (*avgInter*) can be used as a criterion to estimate k for Web page datasets. Our experiments showed that when the *avgInter* for a Web page dataset reaches a threshold of 1.7, the clustering solutions achieve the best results. Compared to other criteria, *avgInter* implies a characteristic for Web page datasets. We then use the threshold as a stopping factor in our clustering process to automatically discover the number of clusters in Web page datasets.

Future work includes the investigation of using our *avgInter* method on datasets from domains other than Web pages. Having the new stopping factor for the Web domain together with the new bi-directional hierarchical clustering algorithm, we have developed a clustering system suitable for mining the Web. We plan to incorporate the new clustering system into our information classification and search engine (Baberwal & Choi, 2004; Choi, 2001; Choi & Dhawan, 2004; Choi & Guo, 2003; Choi & Peng 2004; Yao & Choi, 2003; Yao & Choi, 2005).

ACKNOWLEDGMENT

This research was funded in part by a grant from the Center for Entrepreneurship and Information Technology (CEnIT), Louisiana Tech University.

REFERENCES

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998, June). Automatic subspace clustering for high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD Conference on Management of Data*, Seattle, WA.

Baberwal, S., & Choi, B. (2004, November,). Speeding up keyword search for search engines. *The 3rd IASTED International Conference on Communications, Internet, and Information Technology.* St.Thomas, VI.

Berkhin, P. (2002). *Survey of clustering data mining technique*s. (Tech. Rep.). San Jose, CA: Accrue Software.

Calinski, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics, 3*(1), 1-27.

Choi, B. (2001, October). Making sense of search results by automatic webpage classifications. *WebNet 2001 World Conference on the WWW and Internet*, Orlando, FL.

Choi, B., & Dhawan, R. (2004, September). Agent space architecture for search engines. *The 2004 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, Beijing, China.

Choi, B., & Guo, Q. (2003). Applying semantic links for classifying web pages. *Developments in Applied Artificial Intelligence, IEA/AIE 2003, Lecture Notes in Artificial Intelligence*, 2718, pp.148-153.

Choi, B., & Peng, X. (2004). Dynamic and hierarchical classification of web pages. *Online Information Review, 28*(2), 139-147.

Choi, B., & Yao, Z. (2005). Web page classification. In W. Chu & T. Lin (Eds.), *Foundations and Advances in Data Mining* (pp. 221- 274). Springer-Verag.

Davies, D., & Bouldin D. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 1*(4), 224-227.

Dhillon, I., Fan J., & Guan, Y. (2001). Efficient clustering of very large document collections. In R. Grossman, C. Kamath, P. Kegelmeyer, V. Kumar, & R. Namburu, (Eds.), *Data Mining for Scientific and Engineering Applications* (357-381). Kluwer Academic Publisher.

Dudoit, S., & Fridlyand, J. (2002). A prediction-based resampling method to estimate the number of clusters in a dataset. *Genome Biology, 3*(7), 1-21.

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial database with noise. *International Conference on Knowledge Discovery in Databases and Data Mining*, Portland, OR.

Everitt, B. S., Landua, S., & Leese, M. (2001). *Cluster analysis*. London: Arnold.

Fiduccia, C. M., & Mattheyses, R. M. (1982, June). A Linear Time Heuristic for Improving Network Partitions. In *Proceedings 19th IEEE Design Automation Conference*. Las Vegas, NV.

Guha, S., Rastogi, R. & Shim, K. (1998, June). CURE: A Clustering Algorithm for Large Databases. In *Proceedings of the 1998 ACM*

SIGMOD Conference on Management of Data. Seattle.

Guha, S., Rastogi, R. & Shim, K. (1999b, March). ROCK: A Robust Clustering Algorithm for Categorical Attributes. In *Proceedings of the 15th International Conference on Data Engineering*, Sydney.

Hartigan, J. (1985). Statistical theory in clustering. *Journal of Classification, 2*, 63-76.

Hendrickson, B., & Leland, R. (1993). *A multilevel algorithm for partitioning graphs*. (Tech. Rep. No. SAND93-1301). California: Sandia National Laboratories.

Hinneburg, A., & Keim, D.(1999, September). An optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In *Proceedings of 25th International Conference on Very Large Data Bases*. Scotland, UK.

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data.* Englewood Cliffs, NJ: Prentice-Hall.

Jain, A.,Murty, M., & Flynn, P. (1999). Data clustering: A review. *ACM Computing Surveys, 31*(3), 255-323.

Karypis, G., & Kumar, V. (1999). A fast and high quality multilevel scheme for partitioning rrregular graphs. *SIAM Journal of Scientific Computing, 20*(1), 359-392.

Karypis, G., Han, E.-H., & Kumar, V. (1999). CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer, 32*(8), 68-75.

Kaufman, L., & Rousseeuw, P. J. (1990). *Finding groups in data: An introduction to cluster analysis.* New York: Wiley.

Krzanowski, W., & Lai, Y. (1985). A criterion for determining the number of groups in a dataset using sum of squares clustering. *Biometrics, 44*, 23-34.

Larsen, B., & Aone, C. (1999, August). Fast and effective text mining using linear-time document clustering. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Diego.

Mardia, K.V., Kent, J. T., & Bibby, J. M. (1979). *Multivariate analysis*. New York: Academic Press.

Milligan, G., & Cooper, M. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, *50*, 159-179.

Ng, R., & Han, J. (1994). Efficient and effective clustering methods for spatial data mining. In *Proceedings of the 20th International Conference on Very Large Data Bases*. Santiago, Chile.

Peng, X. (2002). *Automatic Web page classification in a dynamic and hierarchical way*. Master thesis, Louisiana Tech University.

Rajaraman, K., & Pan, H. (2000). Document clustering using 3-tuples. *Pacific Rim International Conference on Artificial Intelligence*

Workshop on Text and Web Mining, Melbourne.

Riboni, D. (2002). Feature selection for web page classification. In *Proceedings of the Workshop of EURASIA-ICT.* Austrian Computer Society.

Rijsbergen, C. J. (1979). *Information retrieval*. London: Butterworths.

Salton, G., & Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, *24*, 513-523.

Sander, J., Ester, M., Kriegel, H. P., & Xu X., (1998). Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery 2*(2), 169-194.

Strehl, A. (2002). *Relationship-based clustering and cluster ensembles for high-dimensional data mining*. Dissertation, The University of Texas at Austin.

Strehl, A. Ghosh, J., & Mooney, R. (2000, July). Impact of similarity measures on Web-page clustering. *AAAI-2000: Workshop of Artificial Intelligence for Web Search,* Austin, TX.

Tantrum, J., Murua, A., & Stuetzle, W. (2002, July). Hierarchical model-based clustering of large datasets through fractionation and refractionation. *The 8th ACM SIGKDD International Conference on Knowledge and Discovery and Data Mining Location*. Edmonton, Canada.

Tibshirani R., Walther, G., & Hastie, T. (2000). *Estimating the number of clusters in a dataset via the gap statistic*. (Tech. Rep.) Palo Alto, California: Stanford University, Department of Bio-statistics.

Yang, Y., & Pedersen, J.(1997, July). A Comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*. Nashville, TN.

Yao Y., & Karypis, G. (2001). *Criterion functions for document clustering: Experiments and analysis* (Tech. Rep.) Minneapolis, MN: University of Minnesota, Department of Computer Science.

Yao, Z. (2004). *Bidirectional hierarchical clustering for Web browsing*. Master thesis, Louisiana Tech University, Ruston.

Yao, Z., & Choi, B. (2003, October). Bidirectional hierarchical clustering for Web mining. *IEEE/WIC International Conference on Web Intelligence*. Halifax, Canada.

Yao, Z., & Choi, B (2005, June). Automatically discovering the number of clusters in Web page datasets. *The 2005 International Conference on Data Mining*, Las Vegas, NV.

Zhang, T., Ramakrishnan, R., & Linvy, M. (1996, May). BIRCH: An efficient data clustering method for very large databases. In *Proceed-*

ings of the ACM SIGMOD Conference on Management of Data. Montreal.

Zhao, Y., & Karypis, G. (1999). Evaluation of hierarchical clustering algorithm for document datasets. *Computing Surveys, 31*(3), 264-323.

Ben Choi, PhD & pilot, is an associate professor in computer science at Louisiana Tech University and is also a pilot of airplanes and helicopters. He has worked in the computer industry as system performance engineer at Lucent Technologies and as a computer consultant. He received his bachelors, master's, and PhD degrees from The Ohio State University. His areas are electrical engineering, computer engineering, and computer science. his works included associative memory, parallel computing, and machine learning. His works include developing software and hardware methods for building intelligent machines and abstracting the universe as a computer.

Zhongmei Yao is currently a PhD student in the Department of Computer Science at Texas A&M University - College Station. She received an master's degree (MS) in computer science from Louisiana Tech University and a bachelor's degree (BS) in engineering from Donghua University, China. Her current research area is computer networking with a focus on Internet-related technologies and protocols.