Web Page Categorization and Feature Selection Using Association Rule and Principal Component Clustering

Jerome Moore, Eui-Hong (Sam) Han,
Daniel Boley, Maria Gini, Robert Gross, Kyle Hastings,
George Karypis, Vipin Kumar, and Bamshad Mobasher

Department of Computer Science and Engineering
University of Minnesota, Minneapolis, MN 55455

Abstract

Clustering techniques have been used by many intelligent software agents in order to retrieve, filter, and categorize documents available on the World Wide Web. Clustering is also useful in extracting salient features of related web documents to automatically formulate queries and search for other similar documents on the Web. Traditional clustering algorithms either use *a priori* knowledge of document structures to define a distance or similarity among these documents, or use probabilistic techniques such as Bayesian classification. Many of these traditional algorithms, however, falter when the dimensionality of the feature space becomes high relative to the size of the document space. In this paper, we introduce two new clustering algorithms that can effectively cluster documents, even in the presence of a very high dimensional feature space. These clustering techniques. which are based on generalizations of graph partitioning, do not require pre-specified ad hoc distance functions, and are capable of automatically discovering document similarities or associations. We conduct several experiments on real Web data using various feature selection heuristics, and compare our clustering schemes to standard distance-based techniques, such as *hierarchical agglomeration clustering*, and Bayesian classification methods, *AutoClass*.

Keywords: clustering, information retrieval, world wide web, association rules, data mining, intelligent software agents.

Web Page Categorization and Feature Selection Using Association Rule and Principal Component Clustering

1 Introduction

The World Wide Web is a vast resource of information and services that continues to grow rapidly. Powerful search engines have been developed to aid in locating unfamiliar documents by category, contents, or subject. Relying on large indexes to documents located on the web, search engines determine the URLs of those documents satisfying a user's query. Often queries return inconsistent search results, with document referrals that meet the search criteria but are of no interest to the user.

While it may not be currently feasible to extract in full the meaning of an HTML document, intelligent software agents have been developed which extract semantic features from the words or structure of an HTML document. These extracted features are then employed to classify and categorize the documents. Clustering offers the advantage that *a priori* knowledge of categories is not needed, so the categorization process is unsupervised. The results of clustering could then be used to automatically formulate queries and search for other similar documents on the Web, or to organize bookmark files, or to construct a user profile.

In this paper, we present two new clustering algorithms based on graph partitioning and compare their performance against more traditional clustering algorithms used in information retrieval. Traditional clustering algorithms either define a distance or similarity among documents, or use probabilistic techniques such as Bayesian classification. Many of these algorithms, however, break down as the size of the document space, and hence, the dimensionality of the corresponding feature space increases. High dimensionality is characteristic of the type of information retrieval applications which are used to filter and categorize hypertext documents on the World Wide Web. In contrast, our partitioning-based algorithms do not rely on pre-specified or ad hoc notions of distance, and they perform well in the presence of a high dimensional space.

In the rest of the paper we describe our new clustering algorithms, the results of a number of experiments using different methods to select a set of features from the documents, and then compare the results of the different clustering algorithms. We show that partitioning clustering methods perform better than traditional distance based clustering.

2 Clustering Methods

Most of the existing approaches to document clustering are based on either probabilistic methods, or distance and similarity measures (see [FBY92]). Distance-based methods such as k-means analysis, hierarchical clustering [JD88] and nearest-neighbor clustering [LF78] use a selected set of words (features) appearing in different documents as the dimensions. Each such feature vector, representing a document, can be viewed as a point in this multi-dimensional space.

There are a number of problems with clustering in a multi-dimensional space using traditional distance- or probability-based methods. First, it is not trivial to define a distance measure in this space. Some words are more frequent in a document than other words. Simple frequency of the occurrence of words is not adequate, as some documents are larger than others. Furthermore, some words may occur frequently across documents. Techniques such as TFIDF [SM83] have been proposed precisely to deal with some of these problems.

Second, the number of all the words in all the documents can be very large. Distance-based schemes generally require the calculation of the mean of document clusters. If the dimensionality is high, then the calculated mean values do not differ significantly from one cluster to the next. Hence the clustering based on these mean values does not always produce very good clusters. Similarly, probabilistic methods such as Bayesian classification used in AutoClass [CS96], do not perform well when the size of the feature space is much larger than the size of the sample set. This type of data distribution seems to be characteristic of document categorization applications on the Web, such as categorizing a bookmark file. Furthermore, the underlying probability models usually assume independence of attributes (features). In many domains, this assumption may be too restrictive. It is possible to reduce the dimensionality by selecting only frequent words from each document, or to use some other method to extract the salient features of each document. However, the number of features collected using these methods still tends to be very large.

Our proposed clustering algorithms which are described in this section are designed to efficiently handle very high dimensional spaces, and furthermore, they do not require the definition of ad hoc distance or similarity metrics. For our evaluation, we compare these algorithms to two well-known methods: Bayesian classification as used by AutoClass [CS96] and *hierarchical agglomeration clustering (HAC)* based on the use of a distance function [DH73].

AutoClass is based on the probabilistic mixture modeling [TSM85], and given a data set it finds maximum parameter values for a specific probability distribution functions of the clusters. The clustering results provide the full description of each cluster in terms of probability distribution of each attributes. The HAC method starts with trivial clusters, each containing one document and iteratively combines smaller clusters that are sufficiently "close" based on a distance metric.

2.1 Association Rule Hypergraph Partitioning Algorithm

In [HKKM97a, HKKM97b], a new method was proposed for clustering related items in transaction-based databases, such as supermarket bar code data, using association rules and hypergraph partitioning. This method first finds set of items that occur frequently together in transactions using association rule discovery methods [AMS+96]. These frequent item sets are then used to group items into hypergraph edges, and a hypergraph partitioning algorithm [KAKS97] is used to find the item clusters. The similarity among items is captured implicitly by the frequent item sets.

In document clustering, each document corresponds to an item and each possible feature corresponds to a transaction. A frequent item sets found using the association rule discovery algorithm corresponds to a set of documents that have a sufficiently large number of features in common. These frequent item sets are mapped into hyperedges in a hypergraph. A hypergraph [Ber76] $H = (V, E)$ consists of a set of vertices V and a set of hyperedges E. A hypergraph is an extension of a graph in the sense that each hyperedge can connect more than two vertices. In this model, the set of vertices V corresponds to the documents, and each hyperedge $e \in E$ corresponds to a set of related documents found. The weight of a hyperedge is calculated as the average confidence [AMS+96] of all the association rules involving the related documents of the hyperedge. For example, if $\{d_1, d_2, d_3\}$ is a frequent item set, then the hypergraph contains a hyperedge that connects d_1, d_2 and d_3. The confidence of an association rule involving documents like $\{d_1, d_2\} \Longrightarrow \{d_3\}$ is the conditional probability that a feature occurs in document d_3 whenever it occurs in d_1 and d_2. Given all the possible association rules with confidence noted as $\{d_1\} \stackrel{0.8}{\Longrightarrow} \{d_2, d_3\}, \{d_1, d_2\} \stackrel{0.4}{\Longrightarrow} \{d_3\}, \{d_1, d_3\} \stackrel{0.6}{\Longrightarrow} \{d_2\}, \{d_2\} \stackrel{0.4}{\Longrightarrow} \{d_1, d_3\}, \{d_2, d_3\} \stackrel{0.8}{\Longrightarrow} \{d_1\}$, and $\{d_3\} \stackrel{0.6}{\Longrightarrow} \{d_1, d_2\}$, the weight of the hyperedge is 0.6.

Next, a hypergraph partitioning algorithm is used to partition the hypergraph such that the weight of the hyperedges that are cut by the partitioning is minimized. Note that by minimizing the hyperedge-cut we essentially minimize the relations that are violated by partitioning the documents into different clusters. Similarly, this method can be applied to word clustering. In this setting, each word corresponds to an item and each document corresponds to a transaction.

This method uses the *Apriori* algorithm [AMS+96] which has been shown to be very efficient in finding frequent item sets and HMETIS [KAKS97] which can partition very large hypergraphs (of size > 100K nodes) in minutes on personal computers.

2.2 Principal Component Partitioning Algorithm

In the principal component algorithm, each document is represented by an N-vector, where N is the number of features to be used for the selection. The values of the components of the N-vector are computed using the traditional TFIDF method [SM83]. The algorithm proceeds by cutting the entire space of documents with a hyperplane passing through the overall arithmetic mean of the documents. The documents are split into two separate groups by means of the hyperplane, and the algorithm is then applied in exactly the same way to split each group in two. This is repeated as many times as desired.

The definition of the hyperplane is based on principal component analysis. The principal components are the eigenvectors of the covariance matrix of documents. Geometrically, we are computing a linear discriminant function with the direction of the discriminant being the principal component defined to be the leading eigenvector of the sample covariance matrix. The motivation behind this selection comes from Karhunen-Loeve analysis, in which the leading eigenvectors of the covariance matrix are treated as the principal features for classification purposes. However, we do not form a lower dimensional representation of the data; instead we simply use the leading vector as the principal component with which to do a cut.

At each stage, the hyperplane itself is constructed using the singular value decomposition (SVD) of a matrix M_t derived from the matrix M whose i,j-th entry is the word count for word i in document j. The matrix M_t is defined by translating all the vectors representing the documents so that their arithmetic mean is at the origin. The SVD of M_t is $USV' = M_t$, where the $'$ symbol denotes transpose, U, V are orthogonal matrices and S is a diagonal matrix with positive diagonal entries in non-increasing order. The diagonal entries of S are called the singular values. The first column $\mathbf{u}_1$ of U is the principal axis of the documents and the orthogonal projection of each document onto this vector is the first column $\mathbf{v}_1$ of V. The hyperplane is normal to $\mathbf{u}_1$, and the side of the hyperplane on which each document lies corresponds to the sign of the corresponding element of $\mathbf{v}_1$.

3 Experimental Results

3.1 Experimental Setup

For the experiments we present here we selected 98 web pages in four broad categories: business and finance, electronic communication and networking, labor, and manufacturing. These pages were downloaded, labeled, and archived. The labeling facilitates an entropy calculation and subsequent references to any page were directed to the archive. This ensures a stable data sample since some pages are fairly dynamic in content.

Experiment	Selection Criteria	Datasets Size	Comments
F1	All words	98x5623	We select all words.
F2	Quantile filtering	98x619	Quantile filtering selects the most freqently occurring words until the accumulated frequencies exceed a threshold of 0.25. In addition, we include all words from the partition that contributes the word that exceeds the threshhold.
F3	Top 20+ words	98x1239	We select the 20 most frequently occuring words and include all words from the partition that contributes the 20th word.
F4	Top 5+ words plus emphasized words	98x1432	We select the top 5+ words as in F3. This list is then augmented with any word that was emphasized in the html document. That is, words appearing in <TITLE>, <H1>, <H2>, <H3>, <I>, <BIG>, <STRONG>, or <EMPHASIZE> tags were added to the list.
F5	Frequent item sets	98x399	We select words from the document word lists that appear in a-priori word clusters. That is, we use an object measure to identify important groups of words.
F6	All words with text frequency > 1	98x2641	We prune the words selected for F1 to exclude those occuring only once.
F7	Top 20+ with text frequency > 1	98x1004	We prune the words selected for F3 to exclude those occuring only once.

Table 1: Setup of experiments.

The word lists from all documents were filtered with a stop-list and "stemmed" using Porter's suffix-stripping algorithm [Por80] as implemented by [Fra92]. We derived seven experiments and clustered the documents using the four algorithms described earlier. Our objective is to reduce the dimensionality of the clustering problem and retain the important features of the documents. The seven experiments are distinguished by further selection rules, as shown in Table 1.

3.2 Evaluation of Clustering Results

Validating clustering algorithms and comparing performance of different algorithms is complex because it is difficult to find an objective measure of quality of clusters. We decided to use entropy as a measure of goodness of the clusters (with the caveat that the best entropy is obtained when each cluster contains exactly one document). We compare the results of the various experiments by comparing their entropy across algorithms (Fig. 1) and across feature selection methods (Fig. 2). Note that the hypergraph partitioning method does not cluster all the documents, so the entropy is computed only for the documents clustered.

Our experiments suggest that clustering methods based on partitioning seem to work best for this type of information retrieval applications, because (1) they do not depend in a choice of a distance function; (2) they do not require calculation of the mean of the clusters, and so the issue of having cluster means very close in space does not apply; (3) they are not sensitive to the dimensionality of the data sets. In particular, both the hypergraph partitioning method and the principle component methods performed much better than the traditional methods regardless of the feature selection criteria used.

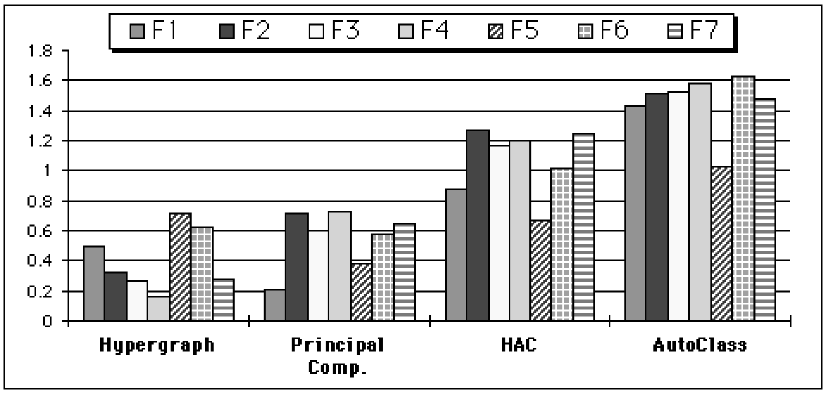

Figure 1: Entropy of different algorithms.

As evident from the experiments the hypergraph partitioning method seems to benefit significantly from careful selection of a small number of representative features from each document. This is partly due to the fact that the item sets are grouped together based on the confidence of the association rules, i.e., the conditional probabilities that features occur in common across the documents. Thus, a larger number of non-relevant words would result in lower confidence rules. The method worked best with F4 (top 5 plus emphasized words) and also well with F3 (top 20 words). The worst results are with F5 (frequent item set). This makes sense, since the hypergraph partitioning includes the computation of the frequent item set as its first set. Applying this method twice may remove too much useful information.

The principal component method works well with large data sets or with filtering that keeps the relevant words. The best results are obtained with F1 (all words), but F5 (frequent item set) is also good. Filtering that removes too much relevant information causes it to perform poorly. We can speculate that F6 (all words that occur more than once) gives good results because most of the relevant features are kept (this is the second largest number of features).

Distance based hierarchical clustering works best with F5 (frequent item set). This is partly due to fact that the dimensionality problem (too few samples given the dimensionality of data) is less acute, but also because, as evident from all of the experiments, finding frequent item (feature) sets as a method for feature extraction seems to effectively isolate the relevant feature for clustering. AutoClass also performs best with F5, but in general, the results are not sufficiently good to make the clusters useful.

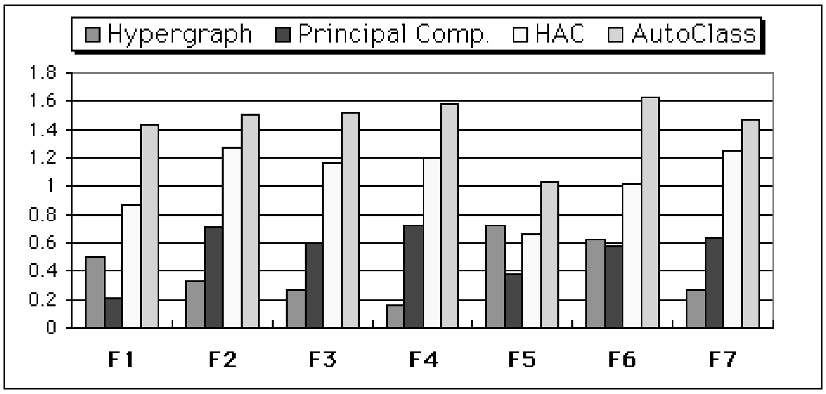

Figure 2: Entropy of different experiments.

To give the reader a feeling on how well the various clustering methods work, we will show in Table 2 some of the document clusters obtained. Full results (including the data sets) are available on the Web at `http://www.cs.umn.edu/~jmoore/wap2.html`.

To provide some more information on the results we show them using a categorization of documents into finer classes within the major categories, though the entropies were computed

Cluster Number	Category	List of Documents
1	B: Finance	1, 15, 16, 17, 18
2	C: Electronic Commerce	76, 77, 78, 79, 80
3	C: Electronic Commerce	11, 12, 13, 20, 22
	C: Networks	6, 22
4	M: Material Processing	45, 46, 49. 50
5	M: General Manufacturing	9, 59, 60, 61, 62, 63, 70
	M: Material Processing	35, 47, 48, 57, 58, 64
6	L: General Labor	24, 25, 26, 27, 28, 29, 38, 39
7	L: Occupational Safety	71, 72, 73, 74, 75
8	L: Affirmative Action	33, 34, 44, 86, 98
9	B: Finance	31, 84, 85
	L: General Labor	97
10	L: General Labor	95, 96
	B: Finance	32
11	M: General Manufacturing	4, 5, 7, 69, 89
12	M: General Manufacturing	87, 88
13	C: Networks	3, 8, 19, 90
	L: General Labor	67, 68
not clustered	C: Networks	2, 21, 30, 83, 91
	L: Affirmative Action	14, 43
	B: Finance	37, 40, 41, 81, 82
	B: Intellectual Property	51, 52, 53, 54, 55, 56, 92, 93
	C: Electronic Commerce	66, 94
	M: General Manufacturing	10, 65
	M: Material Processing	23, 36

Table 2: Results of experiment F4 using hypergraph partitioning. The labels in front of the categories indicate the general classes of the documents (B for Business, C for Communication, L for Labor, M for Manufacturing).

using the coarse labels. To remain unbiased we assigned the finer labels before starting any of the experiments. Table 2 also specifies these sub-categories for each cluster.

4 Related Work

A number of Web agents use various information retrieval techniques [FBY92] and characteristics of open hypertext Web documents to automatically retrieve, filter, and categorize these documents [CH97, BGMZ97, MS96, WP97, WVS+96]. For example, HyPursuit [WVS+96] uses semantic information embedded in link structures as well as document content to create cluster hierarchies of hypertext documents and structure an information space. Pattern recognition methods and word clustering using the Hartigan's K-means partitional clustering algorithm are used in [WP97] to discover salient HTML document features (words) that can be used in finding similar HTML documents on the Web.

Syskill & Webert [PMB96] represents an HTML page with a Boolean feature vector, and then uses naive Bayesian classification to find web pages that are similar, but for only a given single user profile. Also, Balabanovic [BSY95] presents a system that uses a single well-defined profile to find similar web documents for a user. Candidate web pages are located using best-first search, comparing their word vectors against a user profile vector, and returning the highest -scoring pages. A TFIDF scheme is used to calculate the word weights, normalized for document length. The system needs to keep a large dictionary and is limited to one user.

5 Conclusion

In this paper we have presented two new methods for clustering, namely, Association Rule Hypergraph Partitioning and Principal Component Partitioning, that are particularly suitable for the type of information retrieval applications discussed above. These methods do not depend on distance measures, and perform well in high dimensional spaces.

Our experiments suggest that both of these methods perform better than other traditional clustering algorithms regardless of the techniques used for feature selection. In particular, they both perform well, even when all of the features from each document are used in clustering. In addition, the experiments suggest that if the features selected are restricted to those present in frequent item sets, such as those derived from the Apriori Algorithm, then the traditional methods tend to perform better. It is also evident that, the hypergraph partitioning method may perform better, if the features selected include those words emphasized by document authors through the use of HTML tags.

Our future research plans include developing methods for incremental clustering or classification of documents after discovering an initial set of clusters. Furthermore, we plan to investigate the use of clustering techniques proposed here for word clustering. These word clusters can then be used to classify new documents or to search for related documents on the Web.

References

[AMS+96] A. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast discovery of association rules. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI/MIT Press, 1996.

[Ber76] C. Berge. *Graphs and Hypergraphs.* American Elsevier, 1976.

[BGM97] Andrei Z. Broder, Steven C. Glassman, and Mark S. Manasse. Syntactic clustering of the Web. In *6th International World Wide Web Conference*, April 1997. http://proceedings.www6conf.org/HyperNews/get/PAPER205.html.

[BGMZ97] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G Zweig. Syntactic clustering of the web. In *Proc. of 6th International World Wide Web Conference*, 1997.

[BSY95] Marko Balabanovic, Yoav Shoham, and Yeogirl Yun. An adaptive agent for automated Web browsing. *Journal of Visual Communication and Image Representation*, 6(4), 1995. http://www-diglib.stanford.edu/cgi-bin/WP/get/SIDL-WP-1995-0023.

[CH97] C. Chang and C. Hsu. Customizable multi-engine search tool with clustering. In *Proc. of 6th International World Wide Web Conference*, 1997.

[CS96] P. Cheeseman and J. Stutz. Baysian classification (autoclass): Theory and results. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI/MIT Press, 1996.

[DH73] Richard O. Duda and Peter E. Hart. *Pattern Classification and scene analysis.* John Wiley & Sons, 1973.

[FBY92] W. B. Frakes and R. Baeza-Yates. *Information Retrieval Data Structures and Algorithms.* Prentice Hall, Englewood Cliffs, NJ, 1992.

[Fra92] W. B. Frakes. Stemming algorithms. In W. B. Frakes and R. Baeza-Yates, editors, *Information Retrieval Data Structures and Algorithms*, pages 131–160. Prentice Hall, 1992.

[HKKM97a] E.H. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs. In *Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 9–13, Tucson, Arizona, 1997.

[HKKM97b] E.H. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs. Technical Report TR-97-019, Department of Computer Science, University of Minnesota, Minneapolis, 1997.

[JD88] A.K. Jain and R. C. Dubes. *Algorithms for Clustering Data.* Prentice Hall, 1988.

[KAKS97] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Application in VLSI domain. In *Proceedings ACM/IEEE Design Automation Conference*, 1997.

[LF78] S.Y. Lu and K.S. Fu. A sentence-to-sentence clustering procedure for pattern analysis. *IEEE Transactions on Systems, Man and Cybernetics*, 8:381–389, 1978.

[MS96] Y. S. Maarek and I.Z. Ben Shaul. Automatically organizing bookmarks per content. In *Proc. of 5th International World Wide Web Conference*, 1996.

[PMB96] Michael Pazzani, Jack Muramatsu, and Daniel Billsus. Syskill & Webert: Identifying interesting Web sites. In *National Conference on Artificial Intelligence*, pages 54–61, August 1996. http://www.ics.uci.edu/ pazzani/RTF/AAAI.html.

[Por80] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[SM83] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[TSM85] D.M. Titterington, A.F.M. Smith, and U.E. Makov. *Statistical Analysis of Finite Mixture Distributions.* John Wiley & Sons, 1985.

[WP97] Marilyn R. Wulfekuhler and William F. Punch. Finding salient features for personal Web page categories. In *6th International World Wide Web Conference*, April 1997. http://proceedings.www6conf.org/HyperNews/get/PAPER118.html.

[WVS+96] Ron Weiss, Bienvenido Velez, Mark A. Sheldon, Chanathip Nemprempre, Peter Szilagyi, Andrzej Duda, and David K. Gifford. Hypursuit: A hierarchical network search engine that exploits content-link hypertext clustering. In *Seventh ACM Conference on Hypertext*, March 1996. http://paris.lcs.mit.edu/ rweiss/.