2005

Grouping related attributes

Santosh Dawara

Grouping Related Attributes

Santosh Dawara

THESIS

submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

July 2004

APPROVED BY

SUPERVISING COMMITTEE:

Dr. Ankur Teredesai (Chair)

Dr. Edith Hemaspaandra (Reader)

Dr. Roger Gaborski (Observer)

Grouping Related Attributes

by

Santosh Dawara, B.E.

THESIS

Presented to the Faculty of the Graduate School of

Rochester Institute of Technology, Computer Science department

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

THE ROCHESTER INSTITUTE OF TECHNOLOGY

July 2004

Grouping Related Attributes

Santosh Dawara, M.S.
The Rochester Institute of Technology, 2004

Grouping objects that are described by attributes, or clustering is a central notion in data mining. On the other hand, similarity or relationships between attributes themselves is equally important but relatively unexplored. Such groups of attributes are also known as directories, concept hierarchies or topics depending on the underlying data domain. The similarities between the two problems of grouping objects and attributes might suggest that traditional clustering techniques are applicable. This thesis argues that traditional clustering techniques fail to adequately capture the solution we seek. It also explores domain-independent techniques for grouping attributes.

The notion of similarity between attributes and therefore clustering in categorical datasets has not received adequate attention. This issue has seen renewed interest in the knowledge discovery community, spurred on by the requirements of personalization of information and online search technology.

The problem is broken down into (a) quantification of this notion of similarity and (b) the subsequent formation of groups, retaining attributes similar enough in the same group based on metrics that we will attempt to derive. Both aspects of the problem are carefully studied. The thesis also analyzes existing domain-independent approaches to building distance measures, proposing and analyzing

several such measures for quantifying similarity, thereby providing a foundation for future work in grouping relevant attributes.

The theoretical results are supported by experiments carried out on a variety of datasets from the text-mining, web-mining, social networks and transaction analysis domains. The results indicate that traditional clustering solutions are inadequate within this problem framework. They also suggest a direction for the development of distance measures for the quantification of the concept of similarity between categorical attributes.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction: Why group attributes?

The dramatic increase of availability and accessibility of data, in recent times continues unabated, fueled by the presence of the Internet and advances in data storage solutions. It is hard to imagine that a couple of years ago, email services could not offer more than 10MB of storage. A natural consequence of this shift toward a data-centric culture is the sudden urgency for techniques to summarize, organize and personalize information.

It is a trivial task to enumerate a few of the abundant scenarios that play themselves over and over again. Newsgroup users want to be able to view not only the current newsgroup they belong to, but also semantically related communities. Navigating and managing complex, hierarchically organized topic directories manually is no longer a feasible suggestion. Revenue through advertisements is no longer a matter of chance. The degree of personalization of advertisements served over digital mediums is incredibly sophisticated. Functionality to identify and conclude that users who use *spanish* in their emails will probably be interested in dining *mexican* is commonplace across applications. The suggestion that two users of an instant messaging service acknowledge each-other on their *buddy-list* makes us wonder whether they are part of an even larger community or network. Technologies based on such similarities have come a long way; from featuring in research that is mired in natural language understanding; to industrial-strength, efficient and on line applications that perform complex behavior based on simple, automated deductions! All things considered, similarity between attributes is a central notion

in current intelligent information systems. In fact, grouping of similar categorical attributes is an important data mining challenge.

This thesis is a study of this notion of similarity between concepts (or attributes, as is used interchangeably). The objective is to develop efficient methods for quantifying similarity between attributes for categorical datasets. The problems that are of interest, often manifest themselves as categorical data, for example semi-structured email, text files, and transaction data. Based on this ability to group attributes, systems can provide a conceptually organized view, or a directory, concept hierarchies, topics or meta-attributes depending upon the underlying data domain.

Undeniably, the task suggests clustering as a possible solution. However, as the thesis will demonstrate, clustering does not provide the complete answer. A survey of clustering algorithms [26, 23, 28, 22, 41, 27] reveals basic patterns across solutions. The present focus of clustering algorithms is to attempt to cluster objects or observations based on their attributes or features. Clustering algorithms quantify the extent of the similarity between two observations. Often, distance metrics are used, for example, in the euclidean space. The algorithm will then attempt to reduce the error through an objective function such as Entropy [4, 27] or the divergence between two distributions [12]. Similarly, we break the task of grouping related attributes down into two sub-problems, (a) the task of first quantifying the notion of similarity and (b) the subsequent formation of groups, retaining attributes similar enough in the same group based on the derived metrics.

Table 1.1 shows an example of a categorical dataset that describes two makes of cars. The following groupings of the attributes are immediately clear, {Red, Green}; {4-Door, 2-Door}; {v6, v8}. Going back to the earlier breakdown of the problem, how does one quantify the distance between the concepts Red and Green,

Table 1.1: Categorical representation

Automobile	Color	Class	Engine
Toyota Camry	Red	4-Door	v6
Mini-Cooper	Green	2-Door	v8

as compared to the feature '4-Door'? Surely, the two colors are conceptually closer to each other than they are to an attribute that describes how many doors the automobile has. Little research in data mining attempts to define and solve problems like these independent of the underlying domain [20, 19, 41, 16, 4]. Other directly applicable solutions in this space treat the problem subjectively [31, 25, 33, 13] or simply to detect attributes that are similar enough to aid in feature reduction. With this in mind, the work presented is surely novel.

Table 1.2: As binary data

Automobile	Red	Green	4-Door	2-Door	v6	v8
Toyota Camry	1	0	1	0	1	0
Mini-Cooper	0	1	0	1	0	1

Re-examine the same problem in table 1.2. The problem is now to quantify the extent of similarity between the columns for the attributes Red and Green. This thesis explores several conjectures to study similarity between such attributes. It covers both aspects of the problem, inquiring into methods that can not only quantify similarity, but also group similar attributes or concepts. We then go on to propose several distance metrics for quantifying similarity, thereby providing a foundation for future work in grouping relevant attributes. The theoretical results are supported by experiments carried out on a variety of datasets from the text-mining, web-mining, social networks and transaction analysis domains. This emphasizes the

requirement of domain-independent methods. Each of these domains is carefully introduced and explained in chapter 3. Chapter 4 defines the problem, laying down the terminology, a foundation for the evaluation criteria and for further discussion.

Several obstacles exist. Specifically, the high-dimensionality and sparsity of the frequency table are difficult challenges. Also, the resulting clusters are difficult to evaluate. This problem is non-trivial since, evaluation is subjective in nature. Certain clusters maybe accurate in the context of one problem domain while in another, the same cluster may be of poor quality.

Section 4.3 proposes and analyzes novel distance metrics. A set of metrics, some of which are based on information theory are applied and the results are compared to existing metrics and discussed. Section 4.4 presents a set of experiments carried out with a co-clustering or bi-clustering algorithm. Section 4.5 discusses another approach that identifies patterns in classification models to quantify similarity. Chapter 5 summarizes the experiments and results obtained for each of these methods. Chapter 6 concludes the thesis, presenting a final summary of the methods developed and the results obtained.

Data mining algorithms are also required to be robust and insensitive to noise in the input. Finally, performance and scalability is a necessity in the face of very large datasets growing disproportionate to the abilities of the existing knowledge extraction methods. The impact of these aspects on the methods proposed is discussed wherever applicable.

Recent literature in Data Mining has identified streaming data as the next important challenge for data mining algorithms [14]. For example, large corporations handle infinite data streams from their numerous sub-divisions. These data streams could constitute unstructured qualitative market reports, semi-structured day to day email communication and structured sales and demographic data. The thesis

also explores the possibility of building an on line algorithm to tackle such complex data streams while maintaining the most accurate grouping of features.

Chapter 2

Related work

In subsequent sections, existing solutions for grouping related attributes are studied. A brief overview of the solution is presented with adequate references. In all cases, the intention is to discuss the completeness of the solution that is offered. The common theme to all these sections is not just grouping of attributes, but also the reduction of attributes.

This chapter presents several solutions in clustering. This is especially central to the theme of this paper since, a tremendous amount of effort has previously gone into presenting optimal clustering solutions. Clustering methods covered here include those that are based on Information theory. Other novel methods, including dynamical systems, external probes, related research in the area of online algorithms and data streams are also covered.

The contribution of related research also extends to ideas for evaluation of clusters, preprocessing of data sets and formal treatment of the solutions presented. To conclude, this chapter provides adequate justification for the assertion that the problem of grouping attributes has not been completely understood.

2.1 Clustering

Clustering in a single dimension appears to be an applicable solution to the problem. Dataset X maybe viewed as a matrix Z_{nm}. If we were to cluster the transpose of the matrix Z_{nm}, Z^T_{mn}, we would arrive at a clustering of features a_1 to a_m.

Clustering along a single dimension is a well-studied problem and several solutions exist in this space. However, there are serious limitations here. In most practical instances where groups of features are required, $m \ll n$. In general clustering solutions, the algorithm attempts to seek a clustering of the data points described in their various dimensions or in other words described by their features. Clustering of data points is done by applying a distance metric to determine relatively how far apart two data points are from each other, or how far apart a data point is from a cluster center. Thus, it is not surprising that when dealing with high-dimensional data, clustering algorithms are inefficient and inaccurate [12].

Another problem arises when dealing with text databases. Documents are usually described in terms of their keywords. This is a standard approach taken when dealing with Information Retrieval problems. It is popularly known as the *bag of words* approach. Usually, from a set of 100,000 documents, it is not unusual to consider several 10's of thousands of significant keywords. Also, the resulting frequency matrix is extremely sparse in nature [13]. Clustering algorithms perform poorly on such sparse datasets.

There also several clustering algorithms that rely on Information theory and entropy. ENCLUS [8] is prominent in this space. They are incremental methods built for clustering in a single dimension. Thus, existing single dimensional clustering solutions are not easily adapted to grouping attributes.

2.2 Co-clustering

Co-clustering is the simultaneous iterative clustering of both dimensions. It works similar to the standard strategy applied to solve a rubrics cube. Co-clustering also goes by the name of bi-clustering and block clustering.

There exist several implementations of this strategy. One implementation,

Information Theoretic Co-Clustering [12] converges to a local minimum by attempting to measure and minimize the information loss on each iteration in one dimension. The information loss is minimized by showing that the by finding a distribution q that is close to the original distribution in Kullback-Liebler divergence. Information Theoretic Co-clustering has been shown to work well with sparse and high-dimensional datasets. It also provides an analysis of the interplay between the clusters in both dimensions.

However, it has certain limitations in the context of grouping attributes. For example it creates k hard subsets and k needs to be seeded before the method can be applied. It would also be interesting to see if the Hellerstein divergence [5] offers any advantages over the KL divergence. Section 4.4 deals with this method in greater detail.

2.3 Association rule mining

Association rule mining (ARM) [2] is the task of finding k-level associations of the form $a_1 \wedge a_2 \wedge ... \wedge a_k \Rightarrow a_{k+1}$. ARM provides what is known as a *support-confidence* framework. Patterns of the type: *If an arbitrary customer purchases item 'E', he is likely to purchase item 'F' with p% confidence and q% support.* Confidence provides a measure for the conditional probability that given E has been bought, so will item F. Support is a measure of the frequency with which E occurs in an arbitrary transaction.

The support and confidence framework could conceptually be used to determine strong and weak associations between two features. This might be a useful seed to a grouping method. However, not all basket-type problems can be solved successfully within this framework .

For example, it is not possible to capture patterns of the type: *When people*

buy batteries, they are less likely to buy cat-food. [6] This stems from the fact that it is not possible for the framework to capture a negative dependence between two features. With this in mind, it is possible that associations between features might be misleading. Potentially, items tea and coffee may occur with a very high frequency in market transactions. Assume their prior probabilities to be comparable. However, given that a customer buys tea, if $p(c|t) < p(c)$, then there is a negative dependence here. Under these circumstances, while ARM may mistakenly provide the rule: *If a customer buys tea he is likely to buy coffee,* it cannot capture the negative dependence between the two.

Finally, association rules alone cannot capture strong associations between two features that are mutually exclusive in nature but have a strong conceptual link. Brand loyalty maybe cited as an example. People who buy Pepsi will usually not buy Coke although both items are beverages. Just the fact that beverages frequently co-occur in transactions with other similar items indicates that the two are related. Therefore, it is insufficient to look at ARM for grouping attributes.

2.4 External probes

External probes [10, 11] is an external measure of the similarity between two attributes. The measure is determined with respect to a subset of the other features. This subset is termed as the external probes. Given a 0/1 relation r, two attributes a_i and a_j are similar if their sub-relations $\sigma_{a_i=1}(r)$ and $\sigma_{a_j=1}(r)$ are similar.

External probes satisfies some of the basic requirements of a good distance metric [1].

- It is symmetric $d(a_i, a_j) = d(a_j, a_i)$.
- The distance is 0 if and only if the attributes are identical, this requirement

is subject to the application.

- It satisfies the triangle inequality, $d(a_i, a_j) + d(a_j, a_k) \geq d(a_i, a_k)$
- It captures the notion of similarity. As the similarity between two attributes increases, the metric tends to zero

Some of the basic problems that plague association rules are overcome in this case. Take for example the negative dependence discussed earlier. Since we examine all relations where both Pepsi and Coke occur in transactions, with respect to other attributes (for example chips, popcorn, pizza and other items of that nature), we will find clear patterns in transactions indicating that the two beverages are purchased under similar circumstances. For example, when shopping for party supplies, there is a high probability that the customer may buy either beverage. This observation indicates a strong notion of association between the beverages.

There are practical limitations when using this metric. For one it is only applicable on market-basket like and discrete datasets. In the past, agglomerative clustering methods have been used in conjunction with this method to successfully recover clusters [10]. Based on this work, we study the concept in greater detail in section 4.3 and propose new metrics.

2.5 Feature selection and reduction techniques

Feature selection and reduction methods select a subset of the best features or attempt to transform the existing distribution by reducing the feature space. The dominant methods here are domain-driven and do not provide a closed-form solution for measuring similarity. However, they are an alternative to grouping related attributes and therefore deserve a mention.

Examples of such methods are, Principal Component Analysis, Latent Semantic Indexing, [25] and Context-Based Query Expansion [33].

2.6 Structural similarities

Teredesai et. al. [31] have in the past studied attempts to discover feature clusters by utilizing structural similarities between classification models built over every individual feature. However, the methods have shown little resolution into building high quality clusters. Previous work [17, 39] has utilized structural differences and similarities to determine changes in data characteristics. Also, the presence of isomorphic decision trees motivated work in this direction. A substantial part of the effort in this thesis has been to study this direction further. Section 4.5 deals with this topic in greater detail.

2.7 Dynamical systems

STIRR [19] is a recent scalable clustering algorithm that uses non-linear dynamical systems to cluster categorical data. Each categorical attribute value is represented as a weighted vertex in a graph. STIRR maintains multiple copies of such weighted vertices depending on the occurrence of the categorical values in every tuple. These copies $b_1, b_2, \ldots, b_m$, are called basins, the weights on any given vertex may differ across basins. b_1 is called the principal basin; $b_2, b_3, \ldots, b_m$ are called non-principal basins. Starting with a set of weights on all vertices (in all basins), the system is iterated by propagating the weights until a fixed point is reached.

Gibson et. al. argue that when the fixed point is reached, the weights in one or more of the basins $b_2, b_3, \ldots, b_m$ isolate two groups of attribute values on each attribute: the first with large positive weights and the second with small negative weights, and that these groups correspond intuitively to projections of clusters on

the attribute. A strong advantage of this approach is that it is successful in capturing the transitive relationships of similarity well.

STIRR has been critiqued by Ganti et. al. [16] who state that STIRR has two shortcomings: (a) it lacks the automatic grouping of closely related attributes, which is a non-trivial problem and (b) certain classes of clusters cannot be detected by this approach.

2.8 Hoeffding-Chernoff bound

The current trend in data mining methods and research indicates the fascination with algorithms that can function continuously and indefinitely, adapting to concept drift over time. At the same time, the algorithms make sure no information is lost. These methods are known as incremental learning methods.

Domingos et. al. [15, 14] proposed a novel incremental learning method for decision tree construction. The construction algorithm uses the Hoeffding-Chernoff bound [24] to incrementally construct a decision tree over a high-speed data stream. Essentially, the algorithm slides a window over incoming examples while constructing the tree. Each example is assimilated only once. The algorithm proves itself to be scalable, efficient and as accurate as other batch methods for decision tree construction. In terms of coverage, the proposed method could learn on databases several orders of magnitude larger than those feasible with C4.5 [38] with corresponding gains in accuracy.

When clustering attributes, it is sufficient to iteratively consider examples and evolve the clusters [7]. Domingos et. al. have also demonstrated the effectiveness of applying the Hoeffding-Chernoff Bound to k-means clustering [22]. We briefly restate the theory behind the bond. Consider a real-valued random variable x whose range is R. Suppose we have made n independent observations of the

variable, and computed their mean $\bar{x}$. The Hoeffding Bound [24] states that, with probability greater than or equal to $1 - \delta$, the true mean of the variable is within ϵ of $\bar{x}$, where

$$\epsilon = \sqrt{\frac{R^2 \ln(2/\delta)}{2n}}$$

This overview can also be found with Domingos et al. [14].

Chapter 3

Datasets used

3.1 Overview

The data used in the experiments cover a broad range of domains. This was to emphasize the independence of the method from the underlying domain. The datasets presented a good challenge of high-dimensional, sparse, categorical data that required clustering.

1. Synthetic transaction datasets
2. Web-access logs from the RIT, Computer Science department
3. Newsgroup 20
4. IMSCAN dataset

Synthetic transaction datasets were generated using IBM's synthetic data generator [35]. Data sets were preprocessed for inconsistent relations, bad data and uploaded to a relational database for use.

All the datasets enumerated above are categorical in nature. Further sections discuss each dataset, the problem that they represent, preprocessing steps, and the solutions expected. Often, the hypothesis maybe fine-tuned, to outline precise evaluation criteria depending on the domain.

3.2 Semi-structured text data

3.2.1 Newsgroup 20

The newsgroup dataset was downloaded in the form of email communication stored as MIME[1] encoded files. Each file represents communication in the form of a new message, a reply to an earlier message or a message forwarded to the newsgroup. The dataset is divided into 20 forums representing a particular topic. For example, comp.pc.hardware.ibm is a forum reserved for discussions related to IBM PC's. The dataset is maintained by Jason Rennie [34].

For our purposes, a subset of the dataset is sufficient. For example, a subset of 5 groups under the comp.* hierarchy were separated for experiments. The objective is to build a contingency table of documents and the frequency of occurrence of significant keywords in the document. This is known as the *bag-of-words* approach in information retrieval texts. The contingency table shown in table 3.2 is thus

Table 3.1: An example showing keyword presence vectors

Document ID	Keywords
Email 1.	Aardvark, Apparatus, ...
Email 2.	Archetype, Bigotry, ...

similar to a transaction dataset. Yet, it presents additional challenges:

1. It presents a very large number of attributes for a relatively smaller set of documents

2. It is sparse in nature

[1]MIME or Multipurpose Internet email extensions, is an open standard for email encoding in popular use

The hypothesis is, on the basis of the usage of english words in context with other words in a document, it is possible to arrive at groupings of english words which are associated strongly with each other or are similar in terms of their semantic interpretation. For example, within the context of the comp.* newsgroup, a strong association exists between the words *Drive* and the word *Storage*. Also, the words *Annoy* and *Irritate* are synonymous in their usage across contexts.

Table 3.2: The final binary relation

Document ID	Aardvark	Apparatus	Archetype	Bigotry
Email 1.	1	1	0	0
Email 2.	0	0	1	1

3.2.2 Preprocessing

The original messages were presented in the form of normal email communication. Before keywords could be extracted to build a contingency table, the following transformations were applied:

1. Message bodies were extracted from the MIME encoding
2. Stop words, or common english words were removed from the text bodies of the messages
3. Final stage allowed only selected english words to go ahead, thus eliminating proper-nouns and words that are not in the english language
4. An optional stage is to *stem* the words using a standard stemming algorithm to reduce the size of the attribute set

We maintain the stemming stage as an option since it would be informative to avoid domain-specific techniques of feature reduction. The newsgroup dataset was divided and processed yielding the following subsets for further experiments. Refer to table 3.3 for details on the divisions.

Table 3.3: Sub-division of the newsgroup dataset

Newsgroups	Documents	Keywords
alt.atheism	128	40
comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	4891	12633

3.3 Web-access logs

3.3.1 RIT CS Department Weblog data

Table 3.4: An example web-access log

Timestamp	IP Address	URL accessed	HTTP return code
7/11, 19:30:20	192.168.0.2	/sgd/index.html	200
7/11, 19:30:31	127.100.0.3	/cs2/index.html	200
7/11, 19:30:40	192.168.0.2	/sgd/resume.html	200
7/11, 19:30:41	127.100.0.3	/cs2/assign.html	200
7/11, 19:30:42	192.168.0.2	/sgd/adams.html	200
...			
7/11, 19:53:42	192.168.0.2	/faculty/contact.html	200

The web-access logs[2] maybe represented by the tuple *Timestamp, IP Ad-*

[2]Credit is due to Linus Craig, of the RIT Computer Science department for compiling the data

dress, URL accessed, HTTP return code. See table 3.4. The timestamp is the time at which the hit was recorded. The IP address is the unique internet address of the client or the address of the http proxy used by the client. The URL accessed is the identifier for the web page and the return code indicates if the request was successful.

Refer to figure 3.1. Figure (a) represents the frequency of hits from each unique IP address. The highest frequency of hits are from a few IP addresses at the extreme bottom-right of the plot. These IP addresses are attributed to http proxies. Figure (b) represents the distribution of the frequencies of hits over the URL's. The hits are spread evenly across URL's, as is evident from the increased density of data-points within a specific band.

Based on the usage patterns of web-pages accessed within each session, the objective is to group pages that are related to each others. Based on these groups, pages that are strongly associated (within the same group) but are not directly hyper-linked, are suggested to the web master as ranked candidates for linking.

For example, prospective graduate students often navigate from the Computer Science department home page to the home page of the graduate co-ordinator. If the two pages are directly linked, navigating either way is simple and quick. If they are not linked, other indirect paths must be followed to reach the intended destination. These paths may sometimes be unintuitive and therefore hard to discover. However, not all groupings need to suggest simple semantic associations. For example, the frequently asked questions for the laboratory machines need to be linked to programming laboratory classes. A central index page can link to two different laboratory classes. The objective is to uncover such non-trivial patterns by studying the site usage. The hope is that these patterns will suggest links to improve the navigation between the pages.

3.3.2 Preprocessing

It is immediately apparent from the example of the web-access log in table 3.4 that information about each user session is lost. While accesses maybe recorded, it is not immediately apparent where a user-session begins and where it ends. Ideally, each session could be captured to accurately reflect the duration and objective of a individual session. Since we don't have that information, we rely on on simple heuristics to recover sessions from the web-access logs. At the outset, the following constraints are applied:

1. the IP address remains the same throughout the duration of a session
2. a session can last no longer than 60 minutes
3. subsequent hits in a session can be spaced no more than 10 minutes apart

Table 3.5: Sessions reconstructed from web-access logs

Session	IP Address	URL's accessed
1.	192.168.0.2	/sgd/index.html, /sgd/resume.html, /sgd/adams.html
2.	127.100.0.3	/cs2/index.html, /cs2/assign.html
3.	192.168.0.2	/faculty/contact.html

Table 3.6: Weblog dataset statistics after processing

Maximum Session-length	60 seconds
Maximum allowable time between consecutive hits	20 seconds
Valid URL's (m)	183

Table 3.5 shows an example of reconstructed session data from table 3.4.

A trade off is encountered if the IP addresses suspected to represent web proxies are eliminated. The sessions that are built from the other addresses will be

a better reflection of the true sessions. However, the resulting sessions may also be more sparse in nature. This drawback maybe offset by:

1. increasing the duration of a typical web session
2. mining larger web-access logs that span a greater duration of time and therefore clients

For the current set of experiments, the session-length was maintained at a maximum of 60 seconds. Refer to table 3.6 for further information on the pre-processed data.

3.4 Synthetic market-basket data

The synthetic data generator, Quest, could be used to generate transaction or market-basket datasets for use in the experiments. The data generator could be configured to generate a set of n transactions with m items. On an average, only r of the m possible items feature in a transaction.

Table 3.7: Sub-divisions of the Synthetic dataset used for the experiments

Number of transactions (n)	Transaction length (m)	Average number of items per transaction (r)
5	128	40
100	128	40
1000	128	40

The hypothesis is, based on the knowledge of each transaction, we may arrive at a grouping of items that are similar to each other in terms of customer buying behavior. This is not an unreasonable hypothesis, and has featured in mining

market-basket data for association rules and in the study of other transaction related methods.

The ability to generate arbitrarily long datasets provides a good test for scalability. However, the lack of natural labels implies that the dataset loses its appeal in terms of visual interpretation of the results. Thus, this particular problem features less frequently in the experiments undertaken.

3.5 Network event data

3.5.1 IMSCAN dataset

The IMSCAN[3] Dataset presents a good challenge in terms of sparse and high-dimensional categorical data. The dataset is a simple status log of 4878 users, representing the behavior of the users on the AOL instant messenger network for a time period of 25 days. Over this period, all changes in a users status were recorded in the status log along with the time the change took place. An AOL instant messenger user is uniquely referenced using his AIM handle and is allowed to be in only one of 4 possible states $S \in \{online, offline, idle, busy\}$. The user is in the *online* state when he is visible to the other AOL users over the instant messaging network. The user is *offline* when he has logged his handle off the instant messaging network and is no longer available to the other users. The *idle* and *busy* states are simply special cases of the *online* state. The significant difference here is that the user is less likely to be conversing with other AIM users when *idle* or *busy*. Table 3.8 is an example of the tracking data that was recovered.

The dataset presents a unique and rich source of information to reconstruct the social communities that exist within the population of users. There are a few

[3]Credit is due to Resig et al. [37] of the Laboratory for Applied Computing for compiling this dataset

Table 3.8: Sample Tracking Data

Time (s)	User	Status
15	User2	Online
45	User1	Offline
60	User1	Online
130	User4	Away
160	User5	Idle

basic premises. The first premise is, two users can only interact with each other if they are both *online*. The other basic premise is, to be part of the same community, two users must interact with the members of the community. The assumptions are loosely defined here. In further sections, we propose tighter definitions before going on to examine solutions.

3.5.2 Verification through LiveJournal

We collected data from a 3rd party social network (due to its easy accessibility) called LiveJournal. LiveJournal users have the ability to mark other LiveJournal users as being a 'friend' of theirs, thus creating a social network of associated users. These associations form the basis for links within the social network, with each of the users being a vertex. Users can also provide information pertaining to their personal account, such as their Instant Messenger user names. Over 200,000 user names and their associated IM names were collected.

The LiveJournal data served as a third-party data source to help verify some of the associations that were recovered.

Table 3.9: Sample Social Network Data

User	Friends
User1	User3 User5 User7 User18
User2	User3 User4 User8
User3	User1 User2 User19 User30 User31
User4	User2 User6 User19 User20

Table 3.10: Pruned Network Information

Vertices	4878
Edges	309953
Diameter	22
Avg. Shortest Distance	6.009

3.5.3 Preprocessing

In order to prune the dataset, and find an ideal group of users to collect status data on, users with a minimum in and out link degree of 15 were chosen, leaving a group of 4878 users. The smaller group size helped keep the amount of network traffic within a manageable limit, avoiding bandwidth constraints that currently exist within the IM tracking network. On selecting the users, they were then tracked using our IM tracking framework [37] for 25 days. Refer to 3.10 for statistics on the preprocessed data.

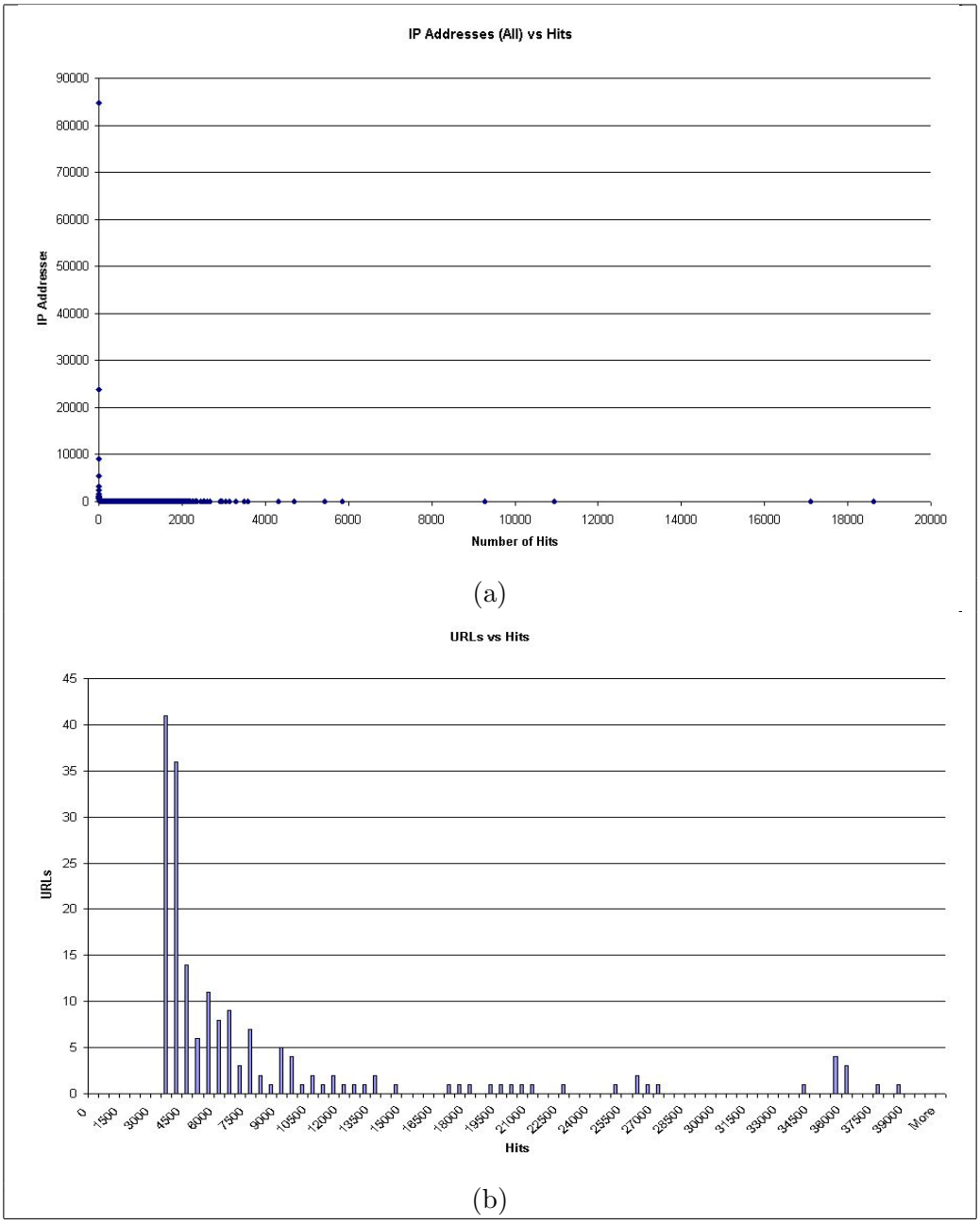

Figure 3.1: (a) Hits versus IP Addresses. (b) URL's versus Hits.

Chapter 4

Grouping related attributes: implementation

The following chapter is central to understanding the problem of grouping attributes and building the theoretical framework for solving the problem. A section is dedicated to a formal presentation of the problem. The formulation shows the uniqueness of this new domain and isolation from the problem of clustering. Remaining sections deal with further exploration of the problem-domain, detailing the fundamentals of distance metrics, internal and external measures and extensions that the thesis proposes. The appeal of each measure is examined qualitatively. The chapter also delves into the theory of co-clustering and the idea of using patterns in the structure of classification models to group attributes. Before that, the chapter details the Data Mining framework and presents its capabilities.

4.1 The data mining framework

4.1.1 System overview

Figure 4.1 provides an abstract view of the primary components that make up the Data mining framework. The framework evolved into a complex system due to the different data storage formats used by the datasets covered in this experiment. Each dataset also presents its own particular challenges. For example, the newsgroup dataset is extremely sparse and high-dimensional in nature. All databases required extensive preprocessing before the algorithms could be applied.

This section's objective is to present the critical aspects of each of the system

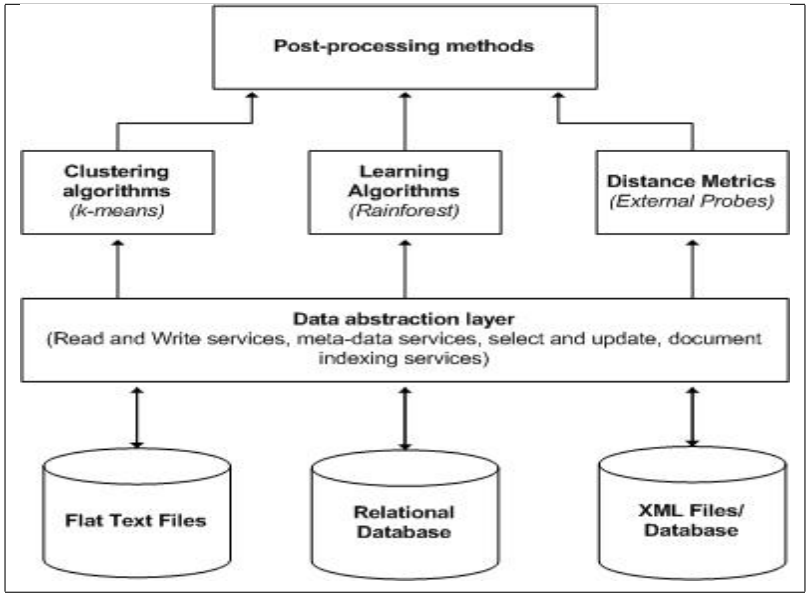

Figure 4.1: An overview of the Data Mining framework

components and to give an accurate picture of the extent of engineering behind the framework.

4.1.2 Preprocessing tools

Preprocessing or data preparation components are a significant part of the framework. Before the different datasets can be consumed by the algorithms, they need to be preprocessed. Preprocessing for the newsgroup dataset (conversion from MIME encoded messages to the final binary relation) and session creation from web-access logs has already been detailed in earlier sections. Preprocessing was not limited to simple conversion of data formats but was also required to filter the data.

All preprocessing components were optimized for performance and could be configured extensively, allowing fine-tuning of the preprocessing step. Preprocessing components share the data access components with the rest of the framework. This

was an advantage from the perspective of rapid development.

4.1.3 Data abstraction layer

The data abstraction layer is designed to abstract the details of data storage from the algorithms. Thus, it is possible to support various types of data storage, including relational databases, XML databases and flat files stored on disk without any modifications to the algorithms that used the data. In addition to read and write support, the data layer also provides query support, data type abstraction and meta-information about the data stored in order to support the functioning of the algorithm.

The data layer also supports high-dimensional data. Conventional databases are designed to support a limited number of attributes per relation. The data layer may be configured to map several attributes to a single byte string of arbitrary length. This is possible since we are dealing exclusively with binary data. This of course requires that the entire relation be transfered from the database to the application before specific tuples can be selected. This reduces the overall efficiency of the system, network throughput being the primary bottleneck.

In terms of underlying data structures, support is not limited to the conventional matrix implementation. Support is also present for storage of data in the CCS (column compressed storage) format [29], a format that stores the dense matrix in its equivalent sparse representation. For sparse datasets, this results in an increase in the efficiency of the algorithms.

The data layer also includes components to efficiently store and index documents [32]. Indices are maintained to map keywords to the documents they occur in. A reverse index from documents to keywords is also maintained. The indices help in efficient searching of documents based on natural language queries and in

the creation of keyword vectors for every document.

4.1.4 Learning classification models

The learning framework offers an environment that can be easily extended to add learning algorithms. A common learning interface is shared across learning algorithms. The data abstraction layer drives[1] these implementations.

The learned classification model, or decision tree can be serialized and stored to disk if configured to do so. The stored model may then be read in by a classification algorithm if the need arises.

4.1.5 Parsing decision trees

The XML representation of the decision trees constructed can be parsed by modules driven by an XML parsing engine. The parsing components are designed to parse and find common patterns between two XML-like documents by looking at their respective substructures.

4.1.6 Clustering data

Clustering methods are built over the data layer, similar to the algorithms for learning classifier models. They expose a common interface and isolate the implementation of the algorithm to promote reuse. The common interface hides the algorithmic details of the underlying implementations.

At this time, the k-means clustering solution, an iterative, clustering algorithm is currently being built using the framework. While the clustering solutions are all dependent on the data layer, the vector representation of each object (or at-

[1] most data mining algorithms are data driven

tributes) can also be preprocessed, for example, using Information theory, to provide a different interpretation of the object's vector before passing it on to the clustering algorithm.

4.2 Problem formulation

Data Set X has m attributes $(a_1, a_2, a_3, \ldots, a_m)$ and a total of n examples. Example $x_i \in X$ and $x_i = (x_{i1}, x_{i2}, x_{i3}, \ldots, x_{im})$. Objective is to partition features a_1 to a_m into groups. Features in the same group describe a common concept. The features themselves may be relatively different from each other. In other words, the problem is to devise a function C_a such that $\hat{a}_j$=$C_a(a_i)$; $\hat{a}_j$ is the set of clusters to which the feature a_i belongs. C_a is the clustering function. Such a function helps us arrive at multiple optimizations of an objective function such as the entropy, or the amount of *disorder* in each cluster.

The problem posed, is NP-complete and moreover, difficult to approximate. In fact, the problem is NP-complete for any distance-function $d(a_i, a_j)$, defined over a pair of attributes a_i and a_j. Therefore, we must resort to heuristics to approximate solutions. In further sections we study domain-independent solutions to this problem. Approximations may exist in several forms. For example, the groups formed above are a similar notion to *soft* clustering solutions. If membership is restricted to only one group, the solution is *hard*. Relatively larger number of algorithms exist in the hard clustering domain. However, hard clusters are further away from the natural groups that we seek.

4.3 Distance measures

4.3.1 An introduction to distance measures

A distance measure allows us to quantify the notion of *similarity* between two concepts. Such a measure is of immense importance in querying databases for example. An oft-cited example is that of a supermarket. Managers would like to determine what products are likely to be similar or related in the customers perspective. An example transaction table is shown in table 4.1. The table shows a set of transactions. A single row is dedicated to each transaction. The columns represent the shelf-items available for purchase. For each transaction, shelf-items that are purchased are represented by the boolean '1', and by the boolean value '0' when ignored.

Table 4.1: Example market transaction table

Transaction	Chips	Mustard	Sausage	Pepsi	Coke	Miller	Bud
01.	1	0	0	0	0	1	0
02.	1	1	1	1	0	1	0
03.	1	0	1	0	1	0	0
04.	0	0	1	0	0	1	0
05.	0	1	1	1	0	0	1
06.	1	1	1	0	0	1	0
07.	1	0	1	1	0	1	0
08.	0	1	1	0	1	0	1
09.	1	0	0	1	0	0	1
10.	0	1	1	0	1	1	1

Patterns can be discovered from transaction data to present ideas of the kind *Pepsi is closer to Coke than it is to Miller* based on distance measures. If we are able to quantify similarities, it is trivial to begin grouping the data. For now, we use our intuitive sense of similarity to create one possible grouping shown in table 4.2. However, other groupings are equally likely. For example, beverages could be

divided further into alcoholic and non-alcoholic.

Table 4.2: One possible set of groups based on the transaction table

Label	Members
Beverages	Pepsi, Coke, Miller, Bud
Snacks	Chips, Sausage, Mustard

These divisions lead to a concept-hierarchy. This is also known as a topic-directory in text mining. Note that not all divisions are neat as implied by the example. Certain shelf items may feature in multiple groups. Also, since there are different possible hierarchies to express the same ideas, the hierarchy is usually set by a domain expert who constrains the search. It is also possible to come up with hierarchy based on simply the information provided by the transaction data and the respective metric.

There are a few fundamental ideas that define the metric. Aggarwal [1] provides an in-depth look at these ideas. Briefly, any metric must successfully capture the perception of similarity and dissimilarity that a user might hold. It should also provide good discriminatory resolution to aid in differentiating concepts that are very similar to every other item. Ideally, a metric should also be statistically sensitive since, data need not be uniformly distributed along any given dimension. This is especially true with large text corpora which display overuse of certain insignificant words. The metric should reflect such statistical biases. The distance function should also be efficiently computable and easily interpretable. The last requirement promotes the applicability of the function in gaining insight and understanding into the application. Therefore, a closed-form solution is extremely desirable.

Defining and evaluating such a metric is not a trivial task. The notion of similarity changes from context to context and must therefore be treated as such to

aid in proper evaluation.

Much of this work is seeded by previous work by Das et. al. [10, 11]. They propose two approaches to computing metrics. The first is, when comparing concepts Pepsi, Chips, we only take into account their respective columns, ignoring other attributes that are part of the transaction. Such metrics are classified as *internal measures*. Additionally, an internal measure is computed from transactions in which $a_i \vee a_j$ is true. Transactions in which $a_i \wedge a_j$ don't feature are ignored.

The basic problem with internal measures is highlighted when attempting to compare two concepts that appear in exclusive transactions. For example, Pepsi and Coke (or Tea and Coffee). It is extremely rare that a transaction will feature both these competing brands. Finding patterns is a lot tougher in this case.

An alternative approach is to view both concepts with respect to the other concepts as well. Such measures are called *external measures*. In the example above, Pepsi and Coke are compared based on transactions in which either of them feature with Chips and other shelf items. Thus, if it is possible to demonstrate that the buying behavior of Pepsi with Chips is the same as Coke with Chips, they are deemed to be similar concepts. Das et. al. have shown that external measures are more accurate and provide more meaningful results.

This chapter proposes new, efficient, external measures, using internal measures as a baseline for comparison. The measures are computed on the Newsgroup 20, IMSCAN and Weblog dataset. The results are discussed and the merits and shortcomings of each method are identified.

Before concluding this section we discuss some basic notation.

Definition 1. Distance: The notion of a distance measure as a single-valued function $d(a_i, a_j)$ which is symmetric ie. $d(a_i, a_j) = d(a_j, a_i)$ for attributes $a_i, a_j \in$

$(a_1, a_2, a_3, ..., a_m)$. The measure maps the inter-attribute distance to real numbers. For additional properties of distance metrics, refer to section 2.4

Definition 2. Sub-relation: Sub-relation σ over relation r is written as $\sigma_{a_i=1}(r)$ where $a_i \in (a_1, a_2, \ldots, a_m)$. It is the enumeration of all tuples with attribute $a_i = 1$.

Sub-relation $\sigma_{a_i=1,a_j=1}(r)$ is the enumeration of all tuples with $a_i = 1 \vee a_j = 1$. For brevity, the sub-relations will be denoted as $\sigma_{a_i}(r)$ and $\sigma_{a_i,a_j}(r)$. The size of the enumerations, or the marginal frequencies will also be denoted as $\sigma_{a_i}(r)$ and $\sigma_{a_i,a_j}(r)$.

4.3.2 Internal measures

Internal measures are computed by focusing on the sub-relations σ_{a_i} and σ_{a_j} where $a_i, a_j \in (a_1, a_2, a_3, \ldots, a_m)$. One possible measure is presented in equation 1

$$d(a_i, a_j) = \frac{\sigma_{a_i}(r) + \sigma_{a_j}(r) - 2 * \sigma_{a_i,a_j}(r)}{\sigma_{a_i}(r) + \sigma_{a_j}(r) - \sigma_{a_i,a_j}(r)} \tag{1}$$

This measure is extremely sensitive to statistical aberrations in the dataset. Even a slight bias towards an attribute is reflected in the metric. Yet, it is easily and efficiently computed and serves as a good baseline for further comparisons.

With most relational databases, the measure in equation 1 can be computed using 3 simple select queries with aggregate operators and no additional memory. With high-dimensional data, it may not be possible to select over the relation since, the rows are stored as byte vectors. Assuming m attributes and n tuples in the relation X, internal distances for all $(m * (m-1))/2$ pairs of attributes may be computed within $O(n * m)$, increasing the space complexity to $O(m * (m-1)/2)$. Also, note that usually $n >> m$. If a sparse matrix representation is used, assume

N to be the number of non-zero (or binary 1's) values. The complexity is revised as $O(N)$. Without the additional memory, the complexity is $O(N * (m * (m-1)/2)))$ in the relation.

4.3.3 External measures

Definition 3. Probe-set P: Is defined as a set of attributes, $P \subseteq (a_1, a_2, a_3, \ldots, a_m)$.

Further, external measures are computed using this probe-set P. For clarity, the probe-set P is expanded as $(p_1, p_2, p_3, \ldots, p_d)$ of size d. One of the important open questions identified by this thesis is what constitutes a good probe-set. We plan to examine this question in greater detail. For now, it suffices to state that in order to compute distance $d(a_i, a_j, P)$, it is possible that, $a_i, a_j \in P$, where $a_i, a_j \in (a_1, a_2, a_3, \ldots, a_m)$. To clearly denote the difference between an internal and external metric, the following notation for external measures is used, for attributes $a_i, a_j \in (a_1, a_2, a_3, \ldots, a_m)$, the distance between them, with respect to Probe-set P, is written as $d(a_i, a_j, P)$.

4.3.3.1 Frequency-based

One possible measure is proposed based on the marginal frequencies of the joint relation between attribute a_i and each of the attributes in the probe-set P.

$$d(a_i, a_j, P) = \sum_{p \in P} \left| \frac{\sigma_{a_i,p}(r)}{\sigma_{a_i}(r)} - \frac{\sigma_{a_j,p}(r)}{\sigma_{a_j}(r)} \right| \tag{2}$$

$$P \subseteq (a_1, a_2, \ldots, a_m); \tag{3}$$

Some observations, the measure is based on what is also known as the confidence or the conditional probabilities for each attribute and an attribute from the

probe-set. $d(a_i, a_j, P)$ is zero when, both attributes a_i, a_j have the same confidence with every attribute in the probe-set indicating a high-degree of similarity. The measure can also be extended to provide a closed-form solution.

4.3.3.2 Internal distance based

Another approach is to define a vector for each attribute and then compute the distance between each vector using fundamental distance metrics. Two issues are raised,

1. The constitution of the vector
2. The choice of the metric

We discuss both issues to some extent in this thesis, with a sharper focus on how to build the vector. However, Dhillon et. al. [3] show that the choice of the metric used is equally important and depends on the underlying dataset.

Definition 4. $\vec{a_i}$: Vector for attribute a, is defined as an ordered enumeration of length d, $(I(a, p_1), I(a, p_2), \ldots, I(a, p_d))$; where $a \in (a_1, a_2, \ldots, a_m)$ and $p_k \in$ probe-set P; and $I(a, p_k)$ is some single-valued function.

$I(a, p_k)$ could be simply the internal distance measure $d(a, p_k)$, see equation 1. Two simple distance metrics, 4, and 5, further illustrate a final solution to computing distance $d(a_i, a_j)$ with respect to probe-set P.

Definition 5. Euclidean distance:

$$d(\vec{a_i}, \vec{a_j}) = \sqrt{\sum_{k=1}^{d} (|a_{ik} - a_{jk}|)^2} \tag{4}$$

Definition 6. Angular distance:

$$\theta(\vec{a_i}, \vec{a_j}) = cos^{-1}\left(\frac{\vec{a_i}.\vec{a_j}}{|a_i|.|a_j|}\right) \tag{5}$$

where $|a_i|, |a_j|$ are the magnitudes of $\vec{a_i}, \vec{a_j}$

4.3.3.3 Information theoretic measures

We also study substituting information gain for $I(a, p_k)$ to construct the vector $\vec{a}$, hypothesizing that information gain will provide accurate and more meaningful results.

Definition 7. Information gain: During construction of a decision tree or a classification tree [38, 18] information gain [9] is used as the objective function to guide the choice attribute to split on at each level. The information gain for each attribute $H(X, C)$ with class-label as C provides the reduction in the uncertainty of C given X. It is a standard for measuring the degree of correlation between two attributes, whether it be positive or negative. It is based on the *entropy* or the amount of *disorder* in the variable X. Refer to [9, 21] for a detailed description.

The gain for attribute $a \in (a_1, a_2, \ldots, a_m)$ is computed for every $p_k \in P$. Some observations on the use of information gain:

1. Information gain is not a symmetrical measure like Mutual Information [9]
2. However, it is a measure of dependence and is non-negative
3. When the two variables are increasingly statistically independent, the gain approaches zero
4. The information gain $I(X, C)$ is a function of the correlation of the attribute X and the class label C with respect to all the other attributes

Algorithm 1 returns the set of vectors for all m attributes. The distance between the two attributes a_i and p_k is determined by using their respective vectors as per equations 4, and 5.

Algorithm 1 getInfoGain, Calculates information gain

Input: Contingency table X, with m attributes $a_1, a_2, \ldots, a_m$,
Probe-set $P = (p_1, p_2, \ldots, p_d)$
Returns vectors: $\vec{a_1}, \vec{a_2}, ..., \vec{a_m}$
Procedure:
for all $a_i \in a_1, a_2, ..., a_m$ **do**
  if $a_i = p_k$ **then**
    $I(a_i, p_k) = 0$
  else
    Calculate information gain I_{a_i,p_k} where $a_i \neq p_k$ and $p_k \in P$
  end if
  $add[\vec{a_i}] \leftarrow I(a_i, p_k)$
end for

4.3.3.4 Complexity

External measures are computationally expensive as our analysis will show. Computation of the metric represented by equation 2 requires $O(n)$ time for a dataset X with n tuples. This requires a proportionate increase in the space complexity which is of the order of $O(m * (m-1))$ to memorize the joint frequencies of all pairs of attributes. Here, m is the number of attributes. The problem could also be reduced to a selection with aggregate operators over the relation when using an RDBMS, thus eliminating the need for extra storage.

Comparatively, computation of the other external measures is inefficient. For the remaining external measures, the cost of computing m vectors is of the order $O(m * d * \beta)$ where d is the size of the probe-set P and β is the complexity of computing each feature. Space complexity is roughly $O(m * d)$, since the vectors

are remembered, thus precluding the necessity to recompute them. This estimate does not include the space required to calculate each feature from relation X.

For information-theoretic measures, $\beta = O(n)$. Based on this analysis and practical observations, computation of this set of external measures is an inefficient process. It is important that the probe-set P, be selected such that $d << m$ for high-dimensional datasets. Further, the algorithms efficiency can be improved by using a sparse matrix representation. In high-dimensional, sparse datasets, like text databases, the number of non-zero values $N << (n*m)$ and $\beta = O(N)$. This results in a dramatic increase in the efficiency of the computation of external measures.

4.3.3.5 Probe-set selection

It is important that some discussion be devoted to the selection of the probe-set P. For the experiments carried out, the probe-set was initially assumed to be the complete set of m attributes. However, based on practical observations and from our complexity analysis of the algorithms, set P could certainly be reduced in size.

It is obvious that the probe-set should be selected so as to ensure that the probes guarantee the best possible coverage of the dataset X. Else, it is possible that our distance measures will fail to provide sufficient resolution into select pairs of attributes. Take for example the metric proposed in equation 2. If for the probe-set P, assume that the attributes a_i and a_j each have a support of 0, with all probe elements $p_i \in P$. Therefore, $d(a_i, a_j, P) = 0$, ie. a_i and a_j are mistakenly flagged as similar.

Reducing the probe-set can provide a tremendous boost to the efficiency of the methods proposed. Also, as the size of the dataset X grows, so will the size of the probe-set. This may quickly become a limiting factor for the scalability of such methods. Comparatively, using information theoretic based measures help

in certain contexts, since measures like entropy are not limited to capturing just positive correlation. Certainly, this requires further investigation.

4.4 Co-clustering

4.4.1 Co-clustering to group AIM users

This section is devoted to the application of co-clustering [26] as a grouping technique on the IMSCAN dataset. Co-clustering methods cluster both dimensions of the relation, thus returning a groupings of rows as well as columns. Cluster membership is reasonably based on similarities in the behavior of any two users on the AIM network. We also examine the argument that associations between users of the AIM network can be recovered by simply looking at their status information. The behavior of the users can be ascertained from the status information. The more inter-linked the behavior of two users, the greater the possibility that they are associated.

By strongly associated, we imply that two users interact with each other on a regular basis, exchanging information with each other over the AIM network as *buddies*. AIM user A is a *buddy* of AIM user B, if and only if, user B has user A on his *buddy list*. AIM users are also part of larger communities or cliques over the AIM network. These communities are connected to each other by virtue of the association between the two *buddies*. A natural formulation is to model this problem as a graph. The vertices represent individual users of the AIM network. Two vertices are connected by an undirected edge if and only if both users represented by the vertices are on each others *buddy list*. We also define and allow directed edges. If user A can see user B on his *buddy list*, and not vice versa, there is a directed edge from user A to user B. Resig et al. [37, 36] analyze the various aspects and problems presented by mining the IMSCAN in detail.

4.4.2 Information theoretic co-clustering

Information-theoretic co-clustering [12] is an implementation of the co-clustering class of algorithms [23]. It is a non-hierarchical and iterative method, similar to the k-means clustering algorithm [22]. Information-theoretic co-clustering attempts to simultaneously cluster both dimensions of the data set. The algorithm provides a clustering function that clusters rows, incorporating column cluster information and vice versa.

Information theoretic co-clustering was the comparative method of choice to meet some of the requirements imposed for determining similarity between users. It satisfies the requirements for both an internal and external method. Internal methods provide a view to compare how similar two users are provided the actions are unbiased and independent; on the other hand, external methods can be used to compute similarity under the dependence constraint. However, a drawback is that the algorithms can only provide hard or non-overlapping clusters. Like k-means, the algorithm also requires to be seeded with an initial value describing the number of groups along each dimension ie. k_{row} and k_{col}. In that respect, it is a sub-optimal solution to our problem[2]. A study of this particular method can establish the following:

1. The methods effectiveness in grouping attributes
2. Provide a baseline for any other attribute grouping algorithm
3. Provide a framework within which the possibility that such grouping algorithms can be studied for enhancement to data streams.

[2]Determining the utility of other clustering techniques is an area for further exploration.

At this time, existing co-clustering implementations have not been extended to data streams. This is a logical step forward in working with co-clustering. Such an enhancement will also provide insight into how the same clusters evolve over time due to steady *drift* in concepts over time.

This study could have tremendous applications in the following areas:

1. Studying trends in web-page content on the Internet
2. Improving navigation of web-sites to match trends in visitor behavior
3. Capturing and countering trends in unsolicited mass and commercial email

4.4.3 Problem formulation

Consider a data set X consisting of N users $(a_1, a_2, a_3, ..., a_N)$, under observation over a period of time (range $[0, T]$). Each tuple represents user-status at time t. The objective is to group users a_1 to a_N into clusters based on similarity in their behavior on the AIM network. Users that fall within the same cluster suggest a link between them. In other words, clustering would provide us with a single or multiple valued function $J(a)$ such that, $J(a_i)$ assigns user a_i to some clusters out of $C_{1..k}$ clusters, where k is the number of possible clusters. Such a clustering scheme is referred to as soft clustering since user a_i is allowed to be a member of more than one cluster. For the purpose of link discovery from clusters, we define an association between two IM users to exist between if an only if:

Definition 1. Users a_i and a_j are considered to be linked if, there exists some cluster C_k, such that, both $a_i, a_j \in C_k$. Thus all users in the same cluster are inter-linked to every other user in that cluster.

The above definition suggests the following requirements of a clustering algorithm to be applicable for link discovery:

1. Since the notion of links is symmetrical, the distance metric used in the clustering has to out put a symmetrical distance measure between two users.

2. The method should be capable of providing a global perspective when determining the relationship between any two users. In other words, the method should be able to leverage the transitive nature of relationships between attributes (here users) as suggested by previous efforts in clustering categorical data [10, 16, 19, 41].

3. The degree of similarity is the least when the behavior of the two users being compared, is unrelated. The degree increases in proportion to an increase in the positive dependence between the behavior of the two users.

4.5 Structural similarities

4.5.1 Of decision trees and production rules

Decision or classification trees are defined and discussed at great lengths in available Data Mining literature [21]. If the reader is unfamiliar with classification trees and their application, they should refer to the relevant resources cited before proceeding.

The following sections refer to a decision tree using capitalized T and a uniquefier, for example T_1. Production rules may be derived from classification trees by following one path from the root of the tree to the leaf. The decision made at every node for the relevant attribute is recorded in the rule. Information about the order in which the attributes are encountered maybe lost. Standard notation is used to describe such rules. For example, the rule $a_1 \wedge \neg a_3 \rightarrow a_2$ refers to records

where attribute $a_1 = 1$ and $a_3 = 0$. With high confidence, the rule predicts that such records have the attribute $a_2 = 1$.

4.5.2 Uncovering patterns in classification models

In previous work [39] similarities between decision trees were utilized to discover clusters of web-pages. The decision trees were built over concepts, or web pages accessed from within web-access sessions. The sessions themselves were collected from web-access logs donated by the RIT, Computer Science department [30]. The technique is discussed in more detail in this section. The objective of this effort was to:

1. Study the application of this technique on the different datasets
2. To investigate its viability further
3. Enhance the existing method by using more sophisticated techniques to present a similarity measure

4.5.3 Definitions

Definition 1. Common examples: An example is common to trees T_1 and T_2 if it follows the same path down both trees down to a leaf node. Here the following rules will have the same set of records satisfying them:
$a_1 \wedge a_3 \rightarrow a_2$, $a_3 \wedge a_2 \rightarrow a_1$

Definition 2. Common path: We define a common-path between two decision trees as the longest common sequence of decisions made on the same attribute values. Refer to figure 4.2.

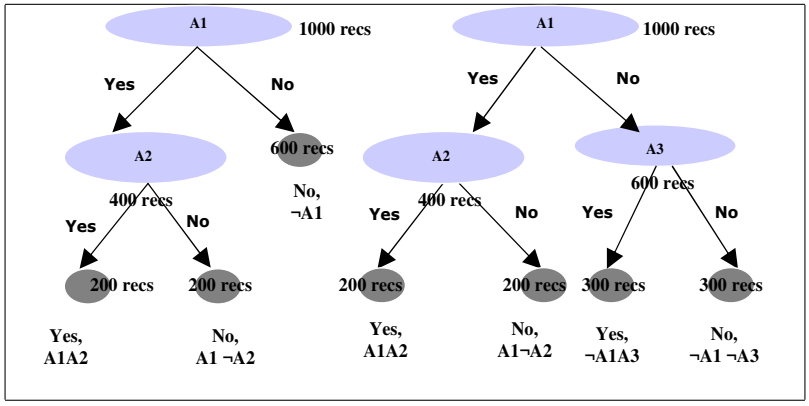

Figure 4.2: An example showing two trees with common paths

Definition 3. Concept tree: The tree construction method is described in greater detail in Teredesai et. al.[39]. This paragraph merely serves as a brief summary of the same. A concept tree here is a decision tree that will classify an example as yes or no with respect to one particular feature. Such m trees for every attribute can be built from the training set by taking each attribute in $a_1, a_2, ...a_m$ as a class-label once.

Definition 4. Jaccard similarity measure: The precise ratio of common records to total records gives the probability of a record R following similar paths in the two trees, with respect to the training dataset. The ratio of common paths to total paths gives us a different measure altogether. Further variations are possible in this scheme:

1. The sequence of the attributes in the path could be ignored
2. Two paths may have common sub-sequences of attributes
3. The leaf node may optionally be included in the path

$$Sim(T_1, T_2) = \frac{|CP(T_1, T_2)|}{|AP(T_1, T_2)|} \tag{6}$$

here,

$CP(T_1, T_2)$ - common paths in trees T_1, T_2

$AP(T_1, T_2)$ - all paths in trees T_1, T_2

So from the example above, the degree of similarity

$$Sim(T_1, T_2) = \frac{4}{(4+2)} = 0.66 \tag{7}$$

This is the same as attempting to measure the intensional agreement between two trees. In other words, what is the probability that a record will be classified down similar paths in the two trees.

On the other hand, extensional agreement describes the possibility that two examples maybe classified the same by both trees even though the examples follow different paths down the tree. There is no notion of extensional disagreement here, since each tree being compared has different class labels than the next.

A basic decision tree can be described by a set of production rules in Conjunctive Normal Form (CNF). If there is a set of production rules that are common between any two trees, the set of examples that will satisfy each member of such a set will therefore be the same. The assumption here is that in order to determine some notion of commonality or similarity in the examples, it is sufficient to look only at the commonality in the production rules.

Take for example two very simple trees for attributes a_1 and a_2. If they are positively correlated (ie. at a distance of 0 from each other), the production rules

that can be derived as a result will be $a_1 \rightarrow a_2$, $!a_1 \rightarrow !a_2$ and $a_2 \rightarrow a_1$, $!a_2 \rightarrow !a_1$. Therefore, a record R will always follow the same path down both trees. The drawback to this method is now clear, negative correlations cannot be identified. Some of the techniques discussed here have been applied and are discussed later.

Definition 5. Node purity: It is not hard to imagine a case where the production rules that are deduced from 2 concept trees, use a disjoint set of predicates:
For example, $a_1 \rightarrow a_2$ and $a_3 \rightarrow a_1$

In this case, assume just three attributes form the dimensions of our search space a_1, a_2, a_3. On extending the rules to their complete depth, $a_1 \wedge a_3 \rightarrow a_2$, $a_1 \wedge !a_3 \rightarrow a_2$ and $a_3 \wedge a_2 \rightarrow a_1$, $a_3 \wedge !a_2 \rightarrow a_1$

Therefore, we need to take into account the measured components [17], ie. the node purity of the concept tree. The node purity differs as we go across trees, due to the nature of the tree construction algorithm which is seeking the most efficient encoding for the data.

4.5.4 Conclusion

Due to constraints and drawback of methods that rely on structural similarities, this sections results are summarized only briefly below. The proposed metrics were computed over the Weblog and Synthetic transaction datasets. The results revealed:

1. This is an unstable method, since there are multiple possible decision trees for the same concept. Therefore, relying on the structure of the model to determine similarity is flawed.

2. The method performs poorly on sparse datasets, due to the increased statistical independence and therefore low commonality, a measure of zero was

returned often by the methods. This does not imply that two attributes are the same. On the contrary, they could be negatively correlated.

3. Any method that requires the construction of m classification models will be computationally intensive. Algorithms immediately become order $O(m^2n)$ solutions, making them inefficient. Here n is the number of tuples in the target relation.

Despite the drawbacks, it is obvious that the visual commonality between two classification trees lends tremendous visual representation potential to this class of methods.

Chapter 5

Experiments and results

5.1 Distance metrics

For the theory behind the experiments, please refer to the section 4.3 on distance metrics. The metrics discussed were used to compute distances between all possible unique pairings of URL's. For the external metrics, the probe-set used was the entire set of m attributes available with each dataset. The experiments were carried out on Intel Pentium 4, 3GHz machines and on Reddwarf, a 4-cpu based Solaris machine with the RIT, CS department. Both sets of machines could allocate up to 1GB of memory to an individual process. These parameters remain unchanged across the experiments.

5.1.1 Weblog dataset

Internal and external distance metrics were calculated over the weblog dataset. Our original hypothesis was that the metrics will predict patterns in visitors' session data indicating sets of similar pages that can be semantically linked, including those pages which are similar and yet feature infrequently in the same session.

What is required are a set of formal methods to analyze these results. This in turn depends on the ability to objectively verify them. A simple approach to validate would be to introduce another variable for every pairing of URL's; The variable should not have been previously used by the metric. For example, we could add a boolean for each pair of URL's indicating if they are linked or not. If we knew

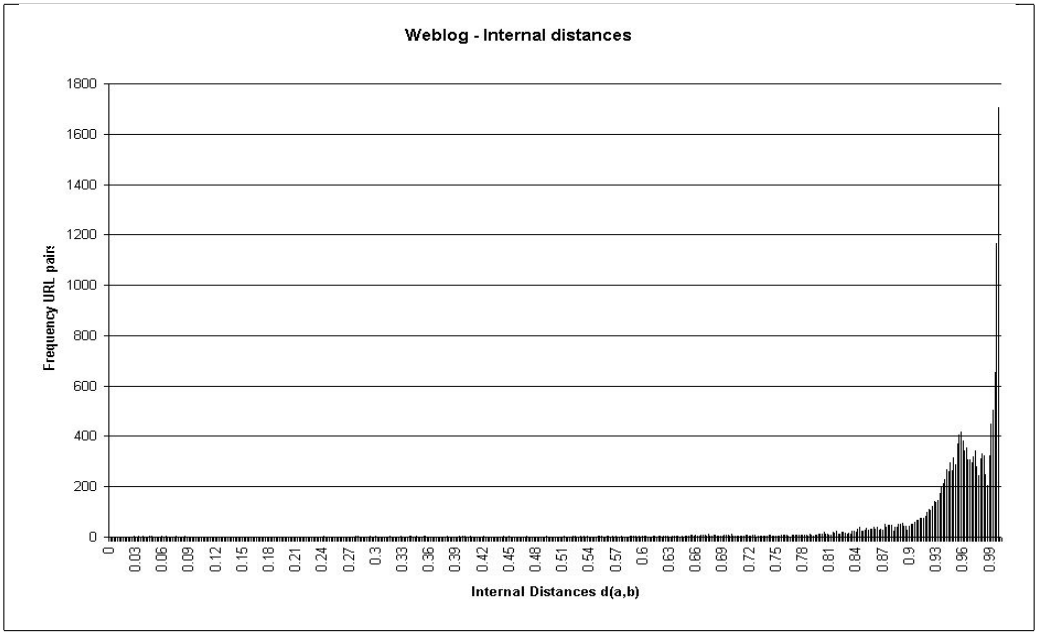

Figure 5.1: Distribution of internal distance $d(a, b)$ for the weblog dataset

which pairs of URL's are actually linked together, we can immediately conclude that probability that those pairs will feature in the same session in sequence will be relatively higher. However, at the time of compilation of this dataset, the linkage information was not recorded. Also, since the time of compilation of this dataset, the RIT CS department website has been reorganized rendering many of the URL's as invalid. Therefore, the linkage information is not entirely recoverable.

Figure 5.1 shows the distribution of internal distances (refer to chapter 4, section 4.3.2) over all possible pairs of web-pages (or URL's). The range for the distances spans [0 to 1] with nearest web-pages close to zero, and furthest neighbors nearer to 1.0. The figure shows a distinct build-up close to the right end of the spectrum. This is possible due to the fact that the distances have been normalized. There is also a short spike of 1706 pairs of URL's at a distance of 1.0 from each other.

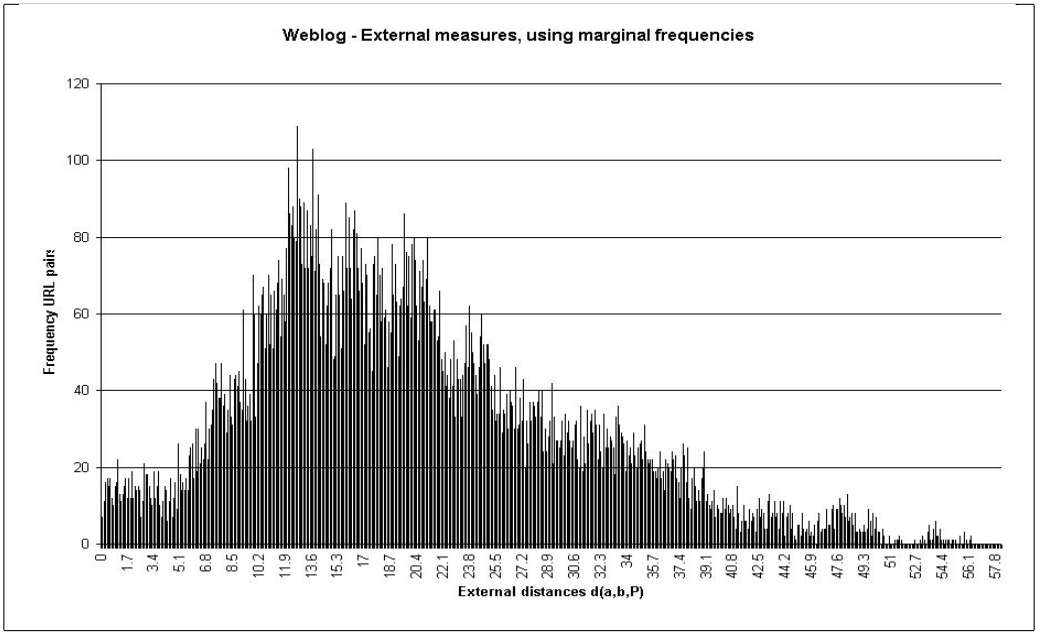

Figure 5.2: Distribution of $d(a, b, P)$ using marginal frequencies for the weblog dataset

This end of the range is populated by pairs of URL's such that, at least one URL of the pair is visited very infrequently. Since this is less of an exception (there are just 16653 possible pairs), the relatively lower resolution causes us to question the desirability of a closed-form solution. A large percentage of the remaining URL's are clustered in the range [0.9 to 0.9999). Due to the lack of formal methods to analyze this range we cannot draw a firm conclusion. By observation, one can draw examples of pages that are semantically related but are calculated to be further apart (see for example: pairs 3 and 4, in table 5.2). We can safely conclude that it is not accurate to operate under assumptions of independence of URL's for this particular problem.

Also presented are the external measures (refer to chapter 4, section 4.3.3) for the same dataset. With the external measures (see figures 5.3 and 5.2) refer to

Table 5.1: Select pairs of URL's

Pair #	URL's a_i, a_j	linked?
1.	/ ats/cs-2002-3/html/skript-frame.html / ats/cs-2002-3/html/skript-1-frame.html	Yes
2.	/ ark/toc.html /deptInfo/index.html	No
3.	/usr/local/jdk/docs/api/java/lang/Object.html / cs1/Labs/01/act1.html	No
4.	/ cs1/Labs/01/act1.html / cs1/Labs/03/act1.html	No
5.	/ atk/Java/Sorting/sorting.html /courses/index.html	No
6.	/ ncs/color/t_convert.html / cs2/Projects/02/proceed.html	No
7.	/ cslab/vi.html / cxv6719/webcam.html	No

the examples in table 5.2. We can go ahead and make a case for a grouping based on the external metrics. Both graphs show clear patterns in the distribution of distances indicating that the patterns will repeat themselves in independent subpopulations or groups. For any grouping method, the objective is to identify centers and the extent of such groups.

5.1.2 Newsgroup 20 and IMSCAN datasets

The Newsgroup 20 and IMSCAN datasets have 12633 and 4087 attributes (refer to tables 3.3 and 3.10 respectively). Due to the CPU and storage intensive nature (refer to discussion on complexity 4.3.3.4) of the external metrics. The experiments to gather results on these two datasets ran several days without completion. At the time of writing this report, the experiments continue to run. Thus, it is not possible to report and discuss the results here.

Table 5.2: Inter-attribute distances

Pair #	Internal distance $d(a_i, a_j)$	Frequency based $d(a_i, a_j, P)$	Information theoretic $d(a_i, a_j, P)$
1.	0.0156	0.2468	4.64E-04
2.	0.9553	3.6463	0.0795
3.	0.9243	19.2053	0.1792
4.	0.8358	7.4967	0.2280
5.	0.9906	9.5442	0.5746
6.	0.9974	48.5628	4.1689
7.	1.0000	12.8784	5.1325

There are two other strategies that could be applied to overcome scale the algorithms for large datasets:

1. Select a smaller probe-set P to reduce the overall running time
2. Sub-divide the datasets further to reduce the number of attributes m

The author intends to pursue both strategies to make available the results for review. The operational realities of these methods reveals the importance of studying the constitution of the probe-set P. The focus of the thesis meant that not enough time could be dedicated to propose methods to select P. In fact, any efficient attempt to select P could avoid relying on the underlying domain characteristics. For example, for the Newsgroup 20 dataset, the inverse document frequency or IDF[40], for each term could be extended to apply in the general case for selection of the probe-set.

5.1.3 Performance

It would be interesting to analyze the behavior of the methods calculating the metrics and time taken with increasing values of m. These results will help verify

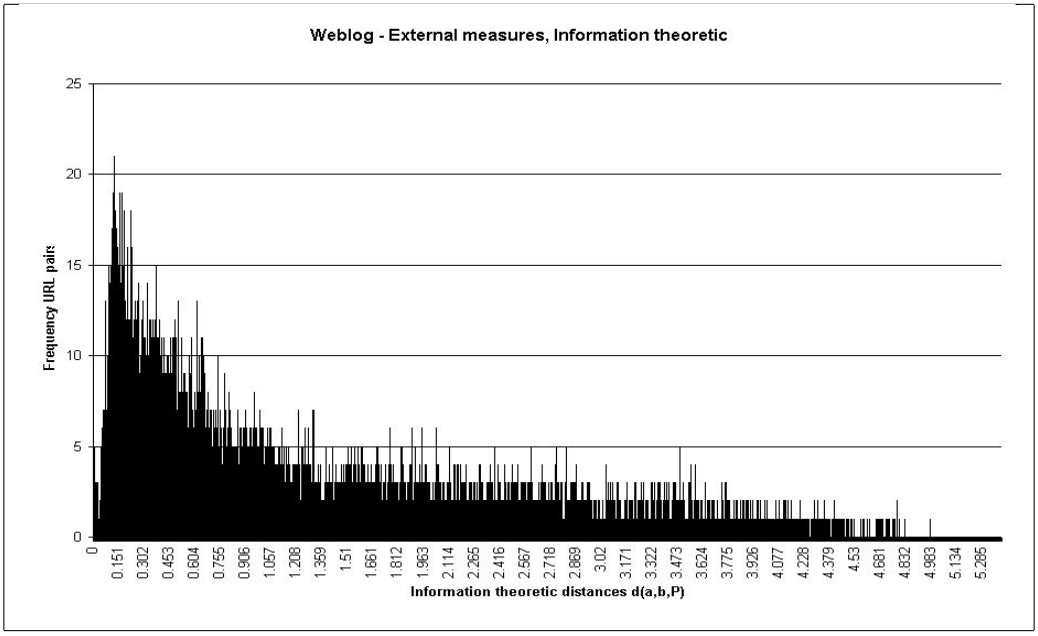

Figure 5.3: Distribution of $d(a, b, P)$ using information theoretic measures and Euclidean distance

the space and time complexity predictions. However, with the unfinished results on the Newsgroup 20 and IMSCAN datasets it is not possible to make any assertions as to the scalability of the methods. As already discussed before the author is working on a method to verify the accuracy of the metrics. From a data mining perspective, the accuracy of a system versus n is also important.

5.2 Grouping AIM users

This section details the experiments and results involving the co-clustering algorithms and the IMSCAN dataset. For related background refer back to section 4.4

The user status log is transformed to highlight any interconnected behavior

between users. This information is used for the further experiments. The strategy adopted to discover associated users is to find patterns in the manner in which users change to the online state and stay in that state for an extended period of time. Based on our knowledge of the instant messaging protocol, a user is only likely to be interacting with another user if they are both in the online state. The assumption that continued interaction may only take place when both users are online is for convenience, and does not accurately reflect the usage of the IM network.

Data set X, the status log is preprocessed to give an intermediate relation $\acute{X}_l$. The preprocessing involves the calculation of a score that reflects the correlation in the behavior of every pair of IM users. The relation $\acute{X}_l$ has dimensions N by N. Each cell x_{ij}, in the relation $\acute{X}_l$ is a measure of the degree to which the behavior of users a_i and a_j correlate. One way to represent the behavior of every pair of users is to define x_{ij} as the number of times users a_i and a_j were found to be online together, every t seconds over the entire time period $[0, T]$ spanning the experiment. Optionally, this score could be weighted by:

1. Reducing it in proportion to the amount of activity in the background. Thus, scores accumulated will be relatively lower if two users are online during peak periods of activity.

2. Reducing it if the two users stay online together for increasing amounts of time. Thus, an event where two users come online together for a short period of time has a greater weighting over a pair that remain online together for relatively greater lengths of time.

Overall, the score is intended to be a function of the probability of a pair of users interacting with each other. However, at this time we want to steer clear of domain-specific techniques.

We then proceed with clustering data set $\acute{X}_l$ to give groups of users for various values of k. It can be observed from the results described in table 5.3 that as the size of k increases the strength of similarity between users declines. This clearly indicates the transient nature of the definition of similarity in this context. Hence, under the assumption of independence of user actions, if direct or internal distances between pairs of users would have been good measures, the memberships would have remained invariant as k is increased. Since the notion of similarity is not correctly captured by the existing clustering methods, we argue that more effort is required to correctly define the notion of similarity (perhaps focusing more on external distances as suggested by Das et. al. [10, 11] and Ganti et. al. [19]) for categorical datasets such as IM status logs. This is necessary to accurately model the relationship between similarity in behavior of a pair of AIM network users and the probability that they are associated. Note that, k_{row} and k_{col} are increased

Table 5.3: Clustering based similarity is dependent on the number of clusters.

k_{row} (rows)	k_{col} (columns)	Memberships discovered
50	50	258,616
100	100	169,292
250	250	94,116
500	500	49,720
750	750	31,080
1000	1000	25,458
1500	1500	14,848

in equal increments, this is an artifact of the algorithm used. Since the algorithm is non-deterministic, by avoiding variations in the number of groups along each dimension for the same trial run, the results remain easily reproducible.

Chapter 6

Conclusion and future work

6.1 Conclusion

This thesis clearly defines an important challenge faced by the knowledge discovery community today. From the description of the datasets in chapter 3, it is evident that preprocessing is an important aspect of identifying patterns in real-world data. This thesis treats it as such, analyzing the possibilities in preprocessing and cleaning the data before advancing ahead. A considerably large amount of effort and time was involved in working with each dataset before the results could be published.

Primarily, through the different real-life datasets and experiments assembled, the importance of grouping attributes, or clustering categorical data has been demonstrated effectively. The thesis clearly shows that problems that depend on such methods abound, inviting and encouraging further research. A by-product of this thesis is also a tremendously flexible and powerful infrastructure to execute experiments.

In terms of algorithmic contribution, the thesis presents innovative strategies. One of these strategies, the strategy of mining patterns in classification models, is certainly novel. The concept of using decision trees for grouping and for studying shifts in data characteristics is not unique [27, 17] but the method proposed to do so, certainly is.

The experiments with metrics and with co-clustering effectively demonstrate

that it is important to consider the transitive and co-dependent nature of variables in order to quantify the notion of similarity. This is an assertion that is central and important to data mining, for solutions that depend on similarity are more the rule than an exception. Based on this assertion the thesis aims to lay a foundation for future work in grouping attributes. The experiments also bring into question the scalability and performance aspects of the metrics proposed. Further efforts are required to investigate the trade offs involved.

6.2 Looking ahead

6.2.1 Methods for grouping attributes

As compared to the baseline internal distance metric, we demonstrate the utility of external measures employing simple histograms. However the true utility of the proposed metrics can only be evaluated by the groups that are created based on these metrics. The next logical step is the development of algorithms that utilize the inter-attribute distance to group attributes. A complete ensemble of such methods will be able to function at various levels, deducing optimal groupings of the attributes. The proposed solutions are NP-complete, restricting them to the use of heuristics to arrive at approximate solutions. With the help of inter-concept distances, it is also possible to model the problem as a weighted undirected graph. The space for graph algorithms is rich and promises several applicable methods to further mine these graphs for patterns.

Efforts will be required in the evaluation of the quality of the resulting groups. Some metrics are proposed for the evaluation of the resulting clusters in the next section. From the nature of the problem, it is clear that a simple objective evaluation is not sufficient, neither is it trivial. Existing strategies employed by clustering algorithms can also be tapped to expedite work in this direction. Being

able to finally arrive at groups also means that the performance of such a method can now be compared to prevalent work in clustering categorical data like ROCK[20], STIRR[19], CACTUS[16], Zhang et. al.[41] and COOLCAT[4].

6.2.2 Evaluation criteria

The performance of a clustering algorithm can be evaluated based on the loss in mutual information after clustering, and also based on the accuracy of the clusters that are recovered from the dataset. The latter depends on foreknowledge of the clusters, its true label, and its membership.

Dhillon et. al. [12] suggest the creation of a confusion matrix if foreknowledge is available. Each entry (i, j) in the confusion matrix represents the number of documents in cluster i that belong to the true class j. Note the use of the term *class* in place of cluster since this is a unsupervised learning method [21].

Another method that can be used is *micro-averaged-precision* [12]. For each class c in the dataset, the precision and recall of the algorithm.

$$P(\hat{y}) = \frac{\Sigma_c(\alpha(c, \hat{y}))}{\Sigma_c(\alpha(c, \hat{y}) + \beta(c, \hat{y}))}, R(\hat{u}) = \frac{\Sigma_c(\alpha(c, \hat{y}))}{\Sigma_c(\alpha(c, \hat{y}) + \gamma(c, \hat{y}))} \quad (1)$$

Here,

$\alpha(c, \hat{y})$: number of attributes classified as c correctly

$\beta(c, \hat{y})$: number of attributes incorrectly classified as c

$\gamma(c, \hat{y})$: number of attributes incorrectly not assigned to c

$P(\hat{y}) = R(\hat{y})$ for uni-labeled data.

These methods represent an objective evaluation of the results. However, it remains to be seen how one can utilize them in determining the quality of groups

of attributes. Some similar, techniques have also been proposed in past work by Barbara et al. [4]. This includes *Significance test on external variables*, that requires the presence of a class attribute. Also proposed is the *category utility function* that attempts to maximize both: the probability that two objects in the same cluster have attribute values in common; and the probability that objects in different clusters have different attribute values.

Another approach that could be adopted is to perform a subjective evaluation of the clusters by asking human subjects to evaluate the quality of the clusters. The problem domain is presented to the subjects and they are then asked to sample and evaluate the quality of the clusters.

6.2.3 Extending the methods to data streams

Online data mining methods mine data streams incrementally. An online clustering method will reevaluate the current clusters based on the new observations that it has seen, without revisiting the old observations it had seen previously. An online algorithm to group attributes is an extremely difficult challenge. Here too, focus persists on the online clustering of records or observations instead of clustering of attributes. Building online algorithms is important from the perspective of evaluation how groups of attributes distill into clusters of attributes that approximates an optimal grouping. It is also a significant goal considering that data mining problems have grown out of the 'toy problem' category presenting important challenges in very large databases.

Bibliography

[1] Charu C. Aggarwal. Towards systematic design of distance functions for data mining applications. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 9–18. ACM Press, 2003.

[2] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

[3] Arindam Banerjee, Srujana Merugu, Inderjit S. Dhillon, and Joydeep Ghosh. Clustering with Bregman divergences. In *Proceedings of the 2004 SIAM International Conference on Data Mining (SDM-04)*, Lake Buena Vista, FL, 2004.

[4] Daniel Barbara, Yi Li, and Julia Couto. Coolcat: an entropy-based algorithm for categorical clustering. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 582–589. ACM Press, 2002.

[5] M. Basseville. Distance measures for signal processing and pattern recognition. In *Signal Processing*, volume 18.4, pages 349–369, December 1989.

[6] Sergey Brin, Rajeev Motwani, and Craig Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, pages 265–276. ACM Press, 1997.

[7] J. Catlett. Megainduction: Machine learning on very large databases. *Unpublished doctoral dissertation, Basser Department of Computer Science, University of Sydney*, 1991.

[8] Chun-Hung Cheng, Ada Waichee Fu, and Yi Zhang. Entropy-based subspace clustering for mining numerical data. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 84–93. ACM Press, 1999.

[9] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, 1991.

[10] Gautam Das, Heikki Mannila, and Pirjo Ronkainen. Similarity of attributes by external probes. In *Knowledge Discovery and Data Mining*, pages 23–29, 1998.

[11] Gautam Das, Heikki Mannila, and Pirjo Ronkainen. Context based similarity measures for categorical databases. In *Principles and Practice of Knowledge Discovery in Databases*, pages 201–210, September 2000.

[12] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM Press, 2003.

[13] Inderjit S. Dhillon and Dharmendra S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1/2):143–175, 2001.

[14] Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge*

discovery and data mining, pages 71–80. ACM Press, 2000.

[15] Pedro Domingos and Geoff Hulten. A general method for scaling up machine learning algorithms and its application to clustering. In *Proc. 18th International Conf. on Machine Learning*, pages 106–113. Morgan Kaufmann, San Francisco, CA, 2001.

[16] Venkatesh Ganti, Johannes Gehrke, and Raghu Ramakrishnan. Cactus: clustering categorical data using summaries. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 73–83. ACM Press, 1999.

[17] Venkatesh Ganti, Johannes Gehrke, and Raghu Ramakrishnan. A framework for measuring changes in data characteristics. In *Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 31 - June 2, 1999, Philadelphia, Pennsylvania*, pages 126–137. ACM Press, 1999.

[18] Johannes Gehrke, Raghu Ramakrishnan, and Venkatesh Ganti. Rainforest - a framework for fast decision tree construction of large datasets. In Ashish Gupta, Oded Shmueli, and Jennifer Widom, editors, *VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases, August 24-27, 1998, New York City, New York, USA*, pages 416–427. Morgan Kaufmann, 1998.

[19] David Gibson, Jon M. Kleinberg, and Prabhakar Raghavan. Clustering categorical data: An approach based on dynamical systems. In Ashish Gupta, Oded Shmueli, and Jennifer Widom, editors, *VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases, August 24-27, 1998, New York City, New York, USA*, pages 311–322. Morgan Kaufmann, 1998.

[20] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345–366, 2000.

[21] Jiawei Han and Micheline Kamber. *Data mining: concepts and techniques*. Morgan Kaufmann Publishers Inc., 2000.

[22] J. A. Hartigan and M. A. Wong. A K-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[23] J.A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, March 1972.

[24] W. Hoeffding. Probability inequalities for sums of bounded random variables. In *Journal of the American Statistical Association*, volume 58, pages 13–30, 1963.

[25] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM Press, 1999.

[26] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[27] Bing Liu, Yiyuan Xia, and Philip S. Yu. Clustering through decision tree construction. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 20–29. ACM Press, 2000.

[28] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[29] National Institute of Standards and Technology. Column compressed storage format. http://math.nist.gov/MatrixMarket/formats.html.

[30] Rochester Institute of Technology. Computer science department. http://www.cs.rit.edu.

[31] Fernando Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of English words. In *31st Annual Meeting of the ACL*, pages 183–190, 1993.

[32] The Apache Jakarta Project. Lucene. http://lucene.apache.org/java/docs/.

[33] Yonggang Qiu and Hans-Peter Frei. Concept-based query expansion. In *Proceedings of SIGIR-93, 16th ACM International Conference on Research and Development in Information Retrieval*, pages 160–169, Pittsburgh, US, 1993.

[34] Jason Rennie. 20 newsgroups data set. http://www.ai.mit.edu/~jrennie/20newsgroups/.

[35] IBM Almadem research. Ibm quest data generator. http://www.almaden.ibm.com/software/quest/Resources/datasets/syndata.ht%ml.

[36] John Resig, Santosh Dawara, Christopher M. Homan, and Ankur Teredesai. Extracting social networks from instant messaging populations. In *Workshop on link analysis and group detection, Knowledge Discovery in Databases, August 2004.*, Seattle, Washington, August 22 2004.

[37] John Resig and Ankur Teredesai. A framework for mining instant messaging services. In *Proceedings of the 2004 SIAM Workshop on Link Analysis, Counter-terrorism, and Privacy*, Lake Buena Vista, Florida, April 24 2004.

[38] Steven L. Salzberg. Book review: *C4.5: Programs for Machine Learning* by J. Ross Quinlan. Morgan Kaufmann Publishers, inc., 1993. *Machine Learning*, 16(3):235–240, 1994.

[39] Ankur Teredesai, Sunil Sharma, and Vineet Chaoji. Auto-indexing web pages using decision trees. *Unpublished project report, Rochester Institute of Technology*, 2003.

[40] T. Tokunaga and M. Iwayama. Text categorization based on weighted inverse document frequency, 1994.

[41] Y. Zhang, A. Fu, C. Cai, and P. Heng. Clustering categorical data. In *In Proceedings of the ICDE*, page 305, 2000.