Exploiting Dataset Similarity for Distributed Mining *

Srinivasan Parthasarathy and Mitsunori Ogihara

Department of Computer Science
University of Rochester
Rochester, NY 14627–0226

{srini,ogihara}@cs.rochester.edu

Abstract. The notion of similarity is an important one in data mining. It can be used to provide useful structural information on data as well as enable clustering. In this paper we present an elegant method for measuring the similarity between homogeneous datasets. The algorithm presented is efficient in storage and scale, has the ability to adjust to time constraints. and can provide the user with likely causes of similarity or dis-similarity.
One potential application of our similarity measure is in the distributed data mining domain. Using the notion of similarity across databases as a distance metric one can generate clusters of similar datasets. Once similar datasets are clustered, each cluster can be independently mined to generate the appropriate rules for a given cluster. The similarity measure is evaluated on a dataset from the Census Bureau, and synthetic datasets from IBM.

1 Introduction

Similarity is a central concept in data mining. Research in this area has primarily progressed along two fronts: object similarity [2, 8, 7] and attribute similarity [5, 9]. The former quantifies how far from each other two objects in the database are while the latter refers to the distance between attributes. Discovering the similarity between objects and attributes enables reduction in dimensions of object profiles as well as provides useful structural information on the hierarchy of attributes.

In this paper we extend this notion of similarity to homogeneous distributed datasets. Discovering the similarity between datasets enables us to perform "meaningful" distributed datamining. Large business organizations with nation-wide and international interests usually rely on a homogeneous distributed database to store their transaction data. This leads to multiple data sources with a common structure. In order to analyze such collection of databases it seems important to cluster them into small number of groups to contrast global trends with local trends rather than apply traditional methods which simply combine them into a single logical resource. A limitation of traditional methods is that the joining is not based on the database characteristics, such as the demographic, economic conditions, and geo-thermal conditions. Mining each database individually is unacceptable as it is likely to generate too many spurious patterns (outliers). We argue for a hybrid solution. First cluster

* This work was supported in part by NSF grants CDA–9401142, CCR–9702466, CCR–9701911, CCR–9725021, INT–9726724, and CCR–9705594; and an external research grant from Digital Equipment Corporation.

the datasets, and then apply the *traditional distributed mining* approach to generate a set of rules for each resulting cluster.

The primary problem with clustering such homogenous datasets is to identify a suitable distance (similarity) metric. The similarity metric depends not only on the kind of mining task being performed but also on the data. Therefore, any measure of similarity should be flexible to both the needs of the task, and data. In this paper we present and evaluate such a similarity metric for distributed association mining. We believe that this metric can be naturally extended to handle other mining tasks such as discretization and sequence mining as well. We then show how one can cluster the database sources based on our similarity metric in an I/O and communication efficient manner. A novelty of our approach to clustering, other than the similarity measure is how we merge datasets without communicating the raw data itself.

The rest of this paper is organized as follows: Section 2 formally defines the problem, and describes our proposed similarity measure. We then present our method for clustering distributed datasets using the aforementioned similarity metric in Section 3. We experimentally validate our approach on real and synthetic datasets in Section 4. Finally, we conclude in Section 5.

2 Similarity Measure

Our similarity measure adopts an idea recently proposed by Das et al [5] for measuring attribute similarity in transaction databases. They propose comparing the attributes in terms of how they are individually correlated with other attributes in the database. The choice of the other attributes (called the probe set) reflects the examiner's viewpoint of relevant attributes to the two. A crucial issue in using this similarity metric is the selection of the probe set. Das *et al.* [5] observed that this choice strongly affects the outcome. However, they do not provide any insight to automating this choice when no apriori knowledge about the data is available. Furthermore, while the approach itself does not limit probe elements to singleton attributes, allowing for complex (boolean) probe elements and computing the similarities across such elements can quickly lead to problems of scale.

We propose to extend this notion of similarity to datasets in the following manner. Our similarity measure compares the datasets in terms of how they are correlated with the attributes in the database. By restricting ourselves to frequently occurring patterns, as probe elements, we can leverage existing solutions (Apriori [3]) for such problems to generate and interactively prune the probe set. This allows us to leverage certain powerful features of associations to handle the limitations described above. First, by using associations as the initial probe set we are able to obtain a "first guess" as to the similarity between two attributes. Second, since efficient solutions for the association problem exist, similarities can be computed rapidly. Third, once this "first guess" is obtained we are able to leverage and extend [1] existing work in interactive (online) association mining [1] to quickly compute similarities under boolean constraints, provide insights into the causes of similarity and dis-similarity, as well as to allow the user to interact and prune the probe space. Finally, we can leverage existing work on sampling to compute the similarity metric accurately and efficiently in a distributed setting.

[1] In addition to the interactions supported in [1] we also support **influential attribute** identification. This interaction basically identifies the (set of) probe attribute(s) that contribute most to the similarity metric.

2.1 Association Mining Concepts

We first provide basic concepts for association mining, following the work of Agrawal *et al.* [3]. Let $\mathcal{I} = \{i_1, i_2, \cdots, i_m\}$ be a set of m distinct *attributes*, also called *items*. A set of items is called an *itemset* where for each nonnegative integer k, an itemset with exactly k items is called a *k-itemset*. A *transaction* is a set of items that has a unique identifier *TID*. The *support* of an itemset A in database $\mathcal{D}$, denoted $\sup_{\mathcal{D}}(A)$, is the percentage of the transactions in in $\mathcal{D}$ containing A as the subset. The itemsets that meet a user specified *minimum support* are referred to as *frequent* itemsets or as *associations*. An *association rule* is an expression of the form $A \Rightarrow B$, where A and B are disjoint itemsets. The *confidence* of an association rule $A \Rightarrow B$ is $\frac{\sup_{\mathcal{D}}(A \cup B)}{\sup_{\mathcal{D}}(A)}$, i.e., the fraction of the datasets containing B over those containing A.

The data mining task for discovering association rules consists of two steps: finding all frequent itemsets (i.e., all associations) and finding all rules whose confidence levels are at least a certain value, the *minimum confidence*. We use our group's ECLAT [11] association mining algorithm to compute the associations.

2.2 Similarity Metric

A measure of similarity between two entities reflects how close they are to one another. Let X and Y be two entities whose similarity we want to measure. We denote $Sim(X, Y)$ to mean the similarity measure between X and Y. Ideally we would like Sim to satisfy the following three properties:

- Identity: $Sim(X, Y) = 1$ corresponds to the fact that the two entities are identical in all respects.
- Distinction: $Sim(X, Y) = 0$ corresponds to the fact that the two entities are distinct in all respects.
- Relative Ordinality: If $Sim(X, Y) > Sim(X, Z)$, then it should imply that X is more similar to Y than it is to Z.

The first two properties bound the range of the measure while the third property ensures that similarities across objects can be meaningfully compared. This last property is particularly useful for clustering purposes.

Now we define our metric. Let A and B respectively be the association sets for a database $\mathcal{D}$ and that for a database $\mathcal{E}$. For an element $x \in A$ (respectively in B), let $\sup_{\mathcal{D}}(x)$ (respectively $\sup_{\mathcal{E}}(x)$) be the frequency of x in $\mathcal{D}$ (respectively in $\mathcal{E}$). Define

$$Sim(A, B) = \frac{\sum_{x \in A \cap B} \max\{0, 1 - \alpha |\sup_{\mathcal{D}}(x) - \sup_{\mathcal{E}}(x)|\}}{\|A \cup B\|}$$

where α is a scaling parameter. The parameter α has the default value of 1 and is to reflect how significance the user view variations in supports are (the higher α is the more influential variations are). For $\alpha = 0$ the similarity measure is identical to $\frac{\|A \cap B\|}{\|A \cup B\|}$, i.e., support variance carries no significance.

2.3 Sampling and Association Rules

The use of sampling for approximate, quick computation of associations has been studied in the literature [10]. While computing the similarity measure, sampling can be used at two levels. First, if generating the associations is expensive (for large datasets) one can sample the dataset and subsequently generate the association set from the sample, resulting in huge I/O savings. Second, if the association sets are

large one can estimate the distance between them by sampling, appropriately modifying the similarity measure presented above. Sampling at this level is particularly useful in a distributed setting when the association sets, which have to be communicated to a common location, are very large.

3 Clustering Datasets

Clustering is commonly used for partitioning data [6]. The clustering technique we adopt is the simple tree clustering. We use the similarity metric of databases defined in Section 2 for as the distance metric for our clustering algorithm. Input to the algorithm is simply the number of clusters in the final result. At the start of the clustering process each database constitutes a unique cluster. Then we repeatedly merge the pair of clusters with the highest similarity and merge the pair into one cluster until there are the desired number of clusters left.

As our similarity metric is based on associations, there is an issue of how to merge their association lattices when two clusters are merged. A solution would be to combine all the datasets and recompute the associations, but this would be time-consuming and involve heavy communication overheads (all the datasets will have to be re-accessed). Another solution would be to intersect the two association lattices and use the intersection as the lattice for the new cluster, but this would be very inaccurate. We take the half-way point of these two extremes.

Suppose we are merging two clusters $\mathcal{D}$ and $\mathcal{E}$, whose association sets are respectively A and B. The value of $\sup_{\mathcal{D}}(x)$ is known only for all $x \in A$ and that of $\sup_{\mathcal{E}}(x)$ is known only for all $x \in B$. The actual support of x in the join of $\mathcal{D}$ and $\mathcal{E}$ is given as

$$\frac{\sup_{\mathcal{D}}(x) \cdot \|\mathcal{D}\| + \sup_{\mathcal{E}}(x) \cdot \|\mathcal{E}\|}{\|\mathcal{D}\| + \|\mathcal{E}\|}.$$

When x does not belong to A or B, we will approximate the unknown sup-value by a "guess" θ [2], which can be specific to the cluster as well as to the association x.

4 Experimental Analysis

In this section we experimentally evaluate our similarity metric[3]. We evaluate the performance and sensitivity of computing this metric using sampling in a distributed setting. We then apply our dataset clustering technique to synthetic datasets from IBM and on a real dataset from the Census Bureau, and evaluate the results obtained.

4.1 Setup

All the experiments (association generation, similarity computation) were performed on a single processor of a DECStation 4100 containing four 600MHz Alpha 21164 processors, with 256MB of memory per processor.

[2] We are evaluating two methods to estimate θ. The strawman is to randomly guess a value between 0 and the minimum support. The second approach is to estimate the support of an item based on the available supports of its subsets.

[3] Due to lack of space we do not detail our experimentation on choice of α.

We used different synthetic databases with size ranging from 3MB to 30MB, which are generated using the procedure described in [3]. These databases mimic the transactions in a retailing environment. Table 1 shows the databases used and their properties. The number of transactions is denoted as $numT$, the average transaction size as T_l, the average maximal potentially frequent itemset size as I, the number of maximal potentially frequent itemsets as $\|L\|$, and the number of items as Size. We refer the reader to [3] for more detail on the database generation.

Database	$numT$	T_l	I	$\|L\|$	Size
D100	100000	8	2000	4	5MB
D200	200000	12	6000	2	12MB
D300	300000	10	4000	3	16MB
D400	400000	10	10000	6	25MB

Table 1. Database properties

The Census data used in this work was derived from the County Business Patterns (State) database from the Census Bureau. Each dataset we derive (dataset per state) from this database contains one transaction per county. Each transaction contains items which highlight information on subnational economic data by industry. Each industry is divided into small, medium and large scale concerns. The original data has numeric data corresponding to number of such concerns occurring in the county. We discretize these numeric values into three categories: high, middle and low. So an item "high-small-agriculture" would correspond to a high number of small agricultural concerns. The resulting set of datasets have as many transactions as counties in the state and a high degree of associativity.

4.2 Sensitivity to Sampling Rate

In Section 2 we mentioned that sampling can be used at two levels to estimate the similarity efficiently in a distributed setting. If association generation proves to be expensive, one can sample the transactions to generate the associations and subsequently use these associations to estimate the similarity accurately. Alternatively, if the number of associations in the lattice are large, one can sample the associations to directly estimate the similarity. We evaluate the impact of using sampling to compute the approximate the similarity metric below.

For this experiment we breakdown the execution time of computing the similarity between two of our databases D300 and D400 under varying sampling rates. The two datasets were located in physically separate locations. We measured the total time to generate the associations for a minimum support of 0.05% (Computing Associations) for both datasets (run in parallel), the time to communicate the associations from one machine (Communication Overhead) to another and the time to compute the similarity metric (Computing Similarity) from these association sets. Transactional sampling influences the computing the associations while association sampling influences the latter two aspects of this experiment. Under association sampling, each processor computes a sample of its association set and sends it to the other, both then compute a part of similarity metric (in parallel). These two values are then merged appropriately, accounting for duplicates in the samples used. While both these sampling levels (transaction and association) could have different

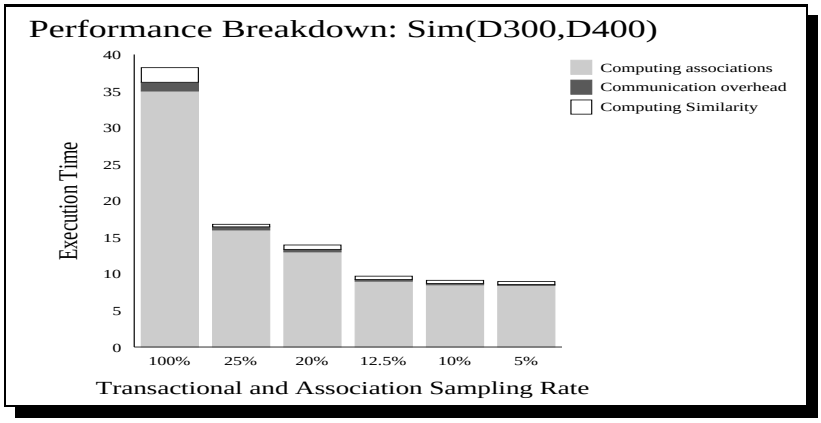

Fig. 1. Sampling Performance

sampling rates, for expository simplicity we chose to set both at a common value. We evaluate the performance under the following sampling rates, 5%, 10%, 12.5%, 20%, and 25%. Figure 1 shows the results from this experiment.

Breaking down the performance it is clear that by using sampling at both levels the performance improves dramatically. For a sampling rate of 10% the time to compute associations goes down by a factor of 4. The communication overhead goes down by a factor of 6 and the time to compute the similarity goes down by a factor of 7. This yields an overall speedup of close to 5. Clearly, the dominant factor in this experiment is computing the associations (85% of total execution time). However, with more traffic in the system, as will be the case when computing the similarity across several datasets (such as in clustering), and when one is modifying the probe set interactively, the communication overhead will play a more dominant role.

The above experiment affirms the performance gains from association and transactional sampling. Next, we evaluate the quality of the similarity metric estimated using such approximation techniques for two minimum support values (0.05% and 0.1%). From Table 2 it is clear that using sampling for estimating the similarity metric can be very accurate (within 2% of the ideal (**S**ampling **R**ate 100%)) for all sampling rates above 5%. We have observed similar results (speedup and accuracy) for the other dataset pairs as well.

Support	SR-100%	SR-25%	SR-20%	SR-10%	SR-5%
0.05%	0.135	0.134	0.136	0.133	0.140
0.1%	0.12	0.12	0.12	0.12	0.115

Table 2. Sampling Accuracy: Sim(D300,D400)

4.3 Synthetic Dataset Clustering

We evaluated the efficacy of clustering homogeneous distributed datasets based on similarity. We used the synthetic datasets described earlier as a start point. We randomly split each of the datasets D100, D200, D300, and D400 into 10 datasets of roughly equal size. For the sake of simplicity in exposition we describe only the

experiment that used only first three subsets from each. We ran a simple tree-based clustering algorithm on these twelve datasets. Figure 2 shows the result. The numbers attached to the joins are the *Sim* metric with $\alpha = 1.0$. Clearly the datasets from the same origin are merged first. Given four as the desired number of clusters (or a merge cutoff of 0.2), the algorithm stops right after executing all the merges depicted by full lines, combining all the children from the same parents into single clusters and leaving apart those from different parents. This experiment illustrates two key points. First, the similarity metric coupled with our merging technique seem to be an efficient yet effective way to cluster datasets. Second, hypothetically speaking, if these 12 datasets were representative of a distributed database, combining all 12 and mining for rules would have destroyed any potentially useful structural rules that could have been found if each cluster were mined independently (our approach).

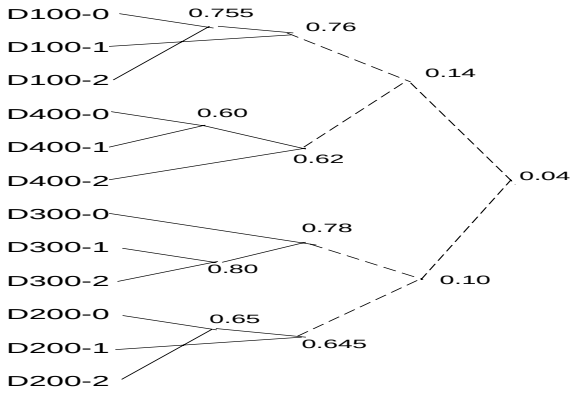

Fig. 2. Dataset Clustering

4.4 Census Dataset Evaluation

Table 3 shows the *Sim* values (with $\alpha = 1.0$) for a subset of the Census data for the year of 1988. As mentioned earlier each dataset corresponds to a state in the US. When asked to break the eight states into four clusters the clustering algorithm returned the clusters [IL, IA, TX], [NY, PA], [FL], and [OR,WA]. On looking at the actual *Sim* values it is clear that NY and PA have a closeted preference for one another IL, IA, and TX have strong preference for one another. OR has a stronger preference for IL, IA and TX, but once IL, IA, and TX were merged it preferred being merged with WA. Interestingly three pairs of neighboring states, i.e., (OR,WA), (IL,IA), and (NY,PA), are found in the same cluster.

An interesting by-play of discretization of the number of industrial concerns into three categories (high, middle and low) is that states with larger counties (area-wise), such as PA, NY and FL tend to have higher associativity (since each county has many items) and thereby tend to have less affinity to states with lower associativity. By probing the similarity between IA and IL further the most influential attribute is found to be agricultural concerns (no surprise there). The reason TX was found to be similar to these states was again due to agricultural concerns, a somewhat surprising result. However, this made sense, when we realized that cattle farming is also grouped

under agricultural concerns! Interestingly, we found that the Census data benefited, performance-wise, from association sampling due its high associativity.

State	IL	NY	PA	FL	TX	OR	WA
IA	0.54	0.01	0.01	0.16	0.44	0.26	0.1
IL		0.02	0.02	0.24	0.52	0.30	0.16
NY			0.31	0.14	0.01	0.04	0.08
PA				0.05	0.01	0.03	0.04
FL					0.24	0.21	0.21
TX						0.32	0.16
OR							0.25

Table 3. Census Dataset: *Sim* Values (support = 20%)

5 Conclusions

In this paper we propose a method to measure the similarity among homogeneous databases and show how one can use this measure to cluster similar datasets to perform meaningful distributed data mining. An interesting feature of our algorithm is the ability to interact via informative querying to identify attributes influencing similarity. Experimental results show that our algorithm can adapt to time constraints by providing quick (speedup of 5-7) and accurate estimates (within 2%) of similarity. We evaluate our work on several datasets, synthetic and real, and show the effectiveness of our techniques. As part of future work we will focus on evaluating and applying dataset clustering to other real world distributed data mining tasks. It seems likely that the notion of similarity introduced here would work well for tasks such as Discretization and Sequence Mining with minor modifications if any. We are also evaluating the effectiveness of the merging criteria described in Section 3.

References

1. C. Aggarwal and P. Yu. Online generation of association rules. In *ICDE*'98.
2. R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In *Foundations of Data Organization and Algorithms*, 1993.
3. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. Inkeri Verkamo. Fast discovery of association rules. In U. Fayyad and et al, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, Menlo Park, CA, 1996.
4. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *20th VLDB Conf.*, 1994.
5. G. Das, H. Mannila, and P. Ronkainen. Similarity of attributes by external probes. In *KDD* 1998.
6. U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The KDD process of rextracing useful information from volumes of data. *Communications of ACM*, 39(11):27–34, 1996.
7. R. Goldman, N.Shivakumar, V. Suresh, and H. Garcia-Molina. Proximity search in databases. In *VLDB Conf.*, 1998.
8. H. Jagadish, A. Mendelzon, and T. Milo. Similarity based queries. In *PODS*, 1995.
9. R. Subramonian. Defining *diff* as a data mining primitive. In *KDD* 1998.
10. H. Toivonen. Sampling large databases for association rules. In *VLDB Conf.*, 1996.
11. M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *KDD*, 1997.