Clustering Distributed Homogeneous Datasets[⋆]

Srinivasan Parthasarathy and Mitsunori Ogihara

Department of Computer Science
University of Rochester
Rochester, NY 14627–0226
{srini,ogihara}@cs.rochester.edu

Abstract. In this paper we present an elegant and effective algorithm for measuring the similarity between homogeneous datasets to enable clustering. Once similar datasets are clustered, each cluster can be independently mined to generate the appropriate rules for a given cluster. 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. The proposed similarity measure is evaluated and validated on real datasets from the Census Bureau, Reuters, and synthetic datasets from IBM.

1 Introduction

Large business organizations, like Sears, with nation-wide or international interests usually rely on a homogeneous distributed database to store their transaction data. This leads to multiple data sources with a common structure. Traditional methods to analyze such distributed databases involve combining the individual databases into a single logical entity before analyzing it. The problem with this approach is that the data contained in each individual database can have totally different characteristics, such as “The customers in the South Carolina store rarely buy winter-related products while those in a store in Maine may buy such items a lot”, leading to a loss of potentially vital information. Another option, mining each database individually is unacceptable as this will likely result in too many spurious patterns (outliers) being generated, wherein it will become harder to decide which patterns are important. Our solution is to first *clusters the datasets*, and then to apply the *traditional distributed mining* approach to generate a set of rules for each resulting cluster.

The primary problem with clustering homogeneous datasets is to identify a suitable distance (similarity) metric. In Section 2 we develop our similarity measure. We then show how one can cluster distributed homogeneous database sources based on our similarity metric in a novel communication efficient manner in Section 3. We then experimentally validate our approach on real and synthetic datasets in Section 4. Finally, we conclude in Section 5.

[⋆] This work was supported in part by NSF grants CDA–9401142, CCR–9702466, CCR–9701911, CCR–9725021, INT–9726724, and CCR–9705594; an NIH grant RO1-AG18231; and an external research grant from Digital Equipment Corporation.

2 Similarity Measure

Similarity is a central concept in data mining. Recently there has been considerable work in defining intuitive and easily computable measures of similarity between complex objects in databases [2,9,8]. The similarity between attributes can also be defined in different ways. An internal measure of similarity is defined purely in terms of the two attributes. The *diff* [11] method is an internal measure where the distance between the probability density function of two attributes is used as a measure of difference between two attributes. An external measure of similarity [5] compares the attributes in terms of how they are individually correlated with other attributes in the database. Das *et al*. [5] show that the choice of the other attributes (called the probe set), reflecting the examiner's viewpoint of relevant attributes to the two, can strongly affect the outcome. However, they do not provide any insight to automating this choice ("first guess") 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 in their approach. We propose an external measure of similarity for homogeneous datasets. 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 (associations), as probe elements, we can leverage existing exact [3] and approximate [13] solutions for such problems to generate the probe sets. Furthermore by using associations as the initial probe set we are able to obtain a "first guess" as to the similarity between two attributes. Also using associations enables one to interactively customize [1] the probe set to inculcate examiner bias and probe for the causes of similarity and dis-similarity.

2.1 Association Mining Concepts

We first provide basic concepts for association mining that are relevant to this paper, 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* [1], 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 $\mathcal{D}$ containing A as the subset. The itemsets that meet a user specified *minimum support* are referred to as *frequent* itemsets or as *associations*. We use our group's ECLAT [12] algorithm to compute the frequent itemsets (*associations*).

2.2 Similarity Metric

Let A and B respectively be the set of associations 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 $\mathcal{E}$). Our metric is:

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

[1] To handle continuous attributes we adopt a novel discretization method not described here due to lack of space.

where α is a scaling parameter. The parameter α has a default value of 1 and can be modified to reflect the significance the user attaches to variations in supports. For $\alpha = 0$ the similarity measure is identical to $\frac{\|A \cap B\|}{\|A \cup B\|}$, i.e., support variance carries no significance. Sim values are bounded and lie in [0,1]. Sim also has the property of *relative ordinalility,* wherein if $Sim(X,Y) > Sim(X,Z)$, then X is more similar to Y than it is to Z. These two properties are essential for our clustering algorithm.

2.3 Interactive Similarity Mining

An important point raised by Das *et al.* [5] is that using a different set of probes could potentially yield a different similarity measure. That property created the need for pruning/constraining the probe space to select appropriate probe sets. In our case the action of modifying the probe set corresponds to modifying the association sets of the input databases, and that is achieved either by modifying the minimum support or by restricting that associations should satisfy certain conditions (Boolean properties over attributes). Supporting such interactions is accomplished by leveraging interactive association mining [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 or dissimilarity of the two datasets.

We define the influence of an attribute a as follows. Let A' be the subset of A containing all associations that include the attribute a. Similarly we define B' with respect to B. Then the influence (INF) and relative influence (RINF) of an attribute a can be defined as follows:

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

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

By definition the attribute which has the largest INF value is the largest contributor to the similarity measure. Also, the attribute which has the lowest INF value is the principle cause for dissimilarity. The user is presented with tuples of the form (INF(a,A,B), RINF(a,A,B)). The RINF values are useful to the user as it conditions the user relative to the size of $A' \cup B'$. As it turns out, these tuples are very useful for probing unexpected/interesting similarity results.

2.4 Sampling and Association Rules

The use of sampling for approximate, quick computation of associations has been studied in the literature [13]. 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 [7]. The technique we adopt for clustering datasets is greedy tree clustering. We use the similarity metric defined in Section 2 in our clustering algorithm. Input to the algorithm is the number of clusters in the final result or a user specified *merge cutoff*. 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 into one cluster until there are the desired number of clusters left or if merging any pair of clusters involves merging clusters that exhibit a *Sim* value that is below the minimal threshold (*merge cutoff*). 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 in both clusters (treating them as one logical entity) and recompute the associations, but this would be time-consuming and involve heavy communication and I/O 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 support of x in the merged cluster is estimated 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 to 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. We first evaluate the performance and sensitivity of computing this metric using sampling, under various support thresholds in a distributed setting. We then evaluate the sensitivity of our metric to choice of α. Finally we demonstrate the efficacy of 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 DECStation 4100s containing four 600MHz Alpha 21164 processors, with 256MB of memory per processor. In order to model distributed market basket data we generated 12 different synthetic datasets ranging from 90MB to 110MB, which are generated adopting 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

[2] Our strawman approach to estimate the value of θ is to randomly guess a value between 0 and the minimum support, since if it does not appear in the set of associations it must have a support less than the minimum support. The second approach, which we evaluate is to estimate the support of an itemset based on the available supports of its subsets.

potentially frequent itemset size as I, the number of maximal potentially frequent itemsets as $\|L\|$, and the number of items was 1000. For each $\|L\|$, I pair (generation pool) we created 3 different (a, b and c) databases by tweaking the database generation program parameters. We refer the reader to [3] for more detail on the database generation.

Database	$numT$	T_l	I	$\|L\|$	Size (range)
$D1_{a,b,c}$	2000000	8-9	2000	4-5	85-90MB
$D2_{a,b,c}$	2000000	10-12	6000	2-4	95-100MB
$D3_{a,b,c}$	2100000	9-10	4000	3-4	100-102MB
$D4_{a,b,c}$	2250000	10-11	10000	6-8	110-120MB

Fig. 1. Database properties

We also use the Reuters-21578 collection [10] to evaluate some aspects of our work. The original data set consists of 21578 articles from the Reuters newswire in 1987. Each article has been tagged with keywords. We created a basket dataset from this collection by representing each news article as a transaction and each keyword being an item. About 1800 articles had no keywords and the average transaction length was between 2 and 3. From this basket dataset we created 12 *country datasets*. Each *country dataset* contains the transactions (country names were among the keywords) that refer to that particular country.

The Census data used in this work was derived from the County Business Patterns (State) database from the Census Bureau. Each dataset (one dataset per state, eight states) 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 attributes (countably infinite) corresponding to number of such concerns occurring in the county which we discretized into three (high, medium, small) categories.

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 $D3_a$ and $D4_a$ 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% and 0.1% (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

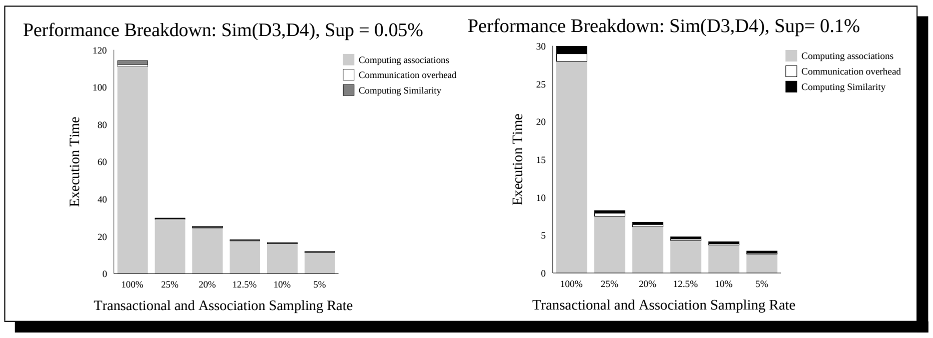

Fig. 2. Sampling Performance

metric (Computing Similarity) from these association sets. Transactional sampling influences the computing of 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 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 2 shows the results from this experiment.

Breaking down the performance it is clear that by using sampling at both levels the performance improves dramatically. A sampling rate of 10% yields an overall speedup of 8. From the figure it is easy to see that the dominant factor in this experiment is computing the associations. However, in a higher latency, lower bandwidth environment (current experiment was in a LAN environment), as will be the case when computing the similarity across distributed datasets interconnected via commodity networks 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 1 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.

4.3 Sensitivity to Support

From Table 1 it should also be clear that the similarity metric is affected by the support. Choosing the “appropriate support” parameter for measuring similarity is an open research problem currently under investigation. The heuristic we use for choosing “appropriate supports” is that the cardinality of the resulting set of associations should lie within a user-specified interval. For the synthetic datasets we used the interval [10000,100000].

Table 1. Sampling Accuracy: Sim($D3_a$,$D4_a$

Support	SR-100%	SR-25%	SR-20%	SR-10%	SR-5%
0.05%	0.136	0.135	0.134	0.135	0.139
0.1%	0.12	0.12	0.12	0.12	0.11

4.4 Similarity Metric: Sensitivity to α

In this section we first evaluate the sensitivity of our similarity metric to the choice of α (alpha). Recall from Section 2 that the choice of α corresponds to the significance the user associates with variation in actual supports.

We evaluate the similarity between three dataset pairs for varying values of α. The results are shown in figure 3A. We wanted to evaluate the robustness to choice of α when two datasets were basically quite similar ($D3_a$,$D3_b$), and when two datasets were basically quite dissimilar ($D3_a$,$D2_a$). We also wanted to see the relative effect between dataset similarity pairs ($D3_a$,$D2_a$ vs $D3_a$,$D4_a$) as α was varied. We varied α from 0.5 to 500 (x axis: log scale). As can be seen from the graph when the datasets are similar the similarity metric is pretty robust to variation in α (up to $\alpha = 200$). When both datasets were dis-similar then they were robust to changes in α when the matches (common itemsets) also had similar support values ($D3_a$,$D4_a$). However, when the matches did not have similar support values ($D2_a$,$D3_a$) increasing α caused a relatively sharp decline in similarity values. This is highlighted by the crossover between the two graphs ($D3_a$,$D4_a$) and ($D2_a$,$D3_a$). This crossover highlights an important fact, *the choice of alpha can affect dataset clustering*. For $\alpha > 100$, dataset $D3_a$ is closer to $D4_a$, but for $\alpha < 100$, $D3_a$ is closer to $D2_a$.

4.5 Dataset Clustering

We now evaluate the efficacy of clustering homogeneous distributed datasets based on similarity. We used the synthetic datasets described earlier as a start point. We ran a simple tree-based clustering algorithm on these twelve datasets. Figure 3B 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 datasets from the same generation pool (as described in Section 4.1) into single clusters and leaving apart those from different pools highlighting the importance of clustering datasets before mining for rules. We next validate our approach on real datasets.

4.6 Census Dataset Evaluation

We asked our algorithm to cluster eight state datasets into four clusters. The clustering algorithm returned the clusters [IL, IA, TX], [NY, PA], [FL], and [OR,WA].

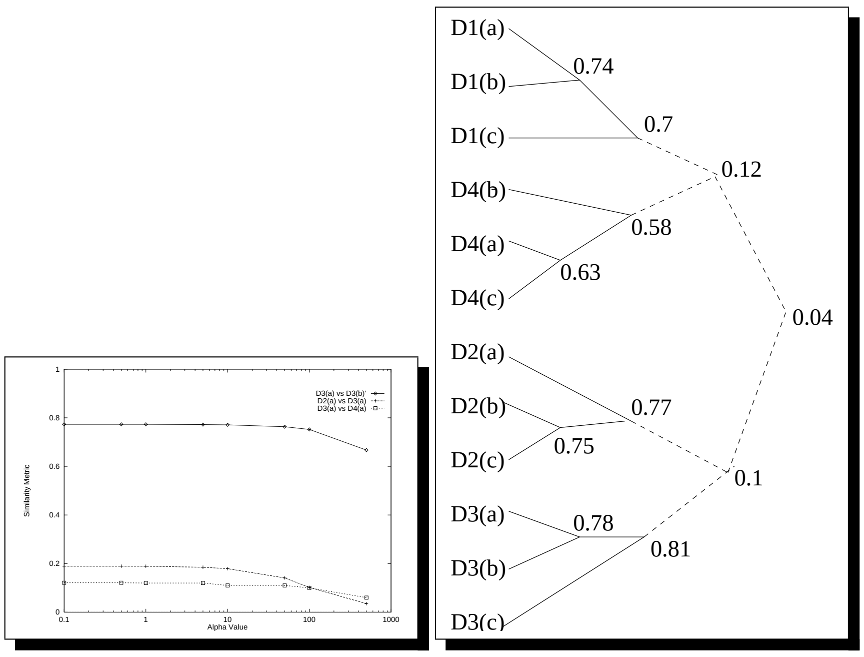

Fig. 3. A: Sensitivity to α B: Clustering Synthetic Datasets

An interesting by-play of our preprocessing step, 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 industries) and thereby tend to have less affinity to states with lower associativity. On probing the similarity between IA, IL and TX [3] the most **influential attribute** is found to be agricultural concerns (no surprise there). This experiment validates our approach as the automatically derived clusters are sensible ones (geographically co-located). Interestingly, we found that the Census data benefits, performance-wise, from association sampling due its high associativity.

4.7 Reuters Dataset Evaluation: Attribute Similarity

Here three clusters were requested by us. The resulting clusters were [India, Pakistan, S. Korea], [USSR, Poland, Argentina], and [Japan, USA, Canada, UK, France, Brazil]. A surprising fact about the first cluster is that India and Pakistan have exactly four transactions in which they co-occur. Our **influential-attribute** algorithm identified that Pakistan and India have common trading partners (UK, France) and are involved in the trade of similar items (various oils and various grains/wheat). In the second cluster the inclusion of Argentina is not intuitive. Probing the cause for this our algorithm was able

[3] Cattle farming is also grouped under agricultural concerns in the Census data.

to identify [4] that although Argentina and Poland, have no co-occurring transactions, they are involved in the trade for similar items (wheat, grain, oilseed, etc.) resulting in the strong Sim value. Similarly the similarity between Argentina and USSR was found to be due to agricultural trade. The third cluster essentially consists of advanced countries. Most of which include Brazil as a strong trading partner resulting in that countries presence in the cluster.

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 4-8) and accurate estimates (within 2%) of similarity. We evaluated our work on several datasets, synthetic and real, and show the effectiveness of our techniques.

Our similarity metric is sensitive to the support and the choice of α. As part of ongoing work we are investigating whether we can automate/guide the choice of these parameters subject to certain user/domain constraints. We are also evaluating the effectiveness of the current merging criteria, and exploring other criteria, as described in Section 3. As part of future work we will focus on evaluating and applying dataset clustering to other real world distributed data mining tasks.

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 *VLDB*'94.
5. G. Das *et al*. Similarity of attributes by external probes. In *KDD* 1998.
6. L. Devroye. A course in density estimation. In *Birkhauser:Boston MA*, 1987.
7. 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.
8. R. Goldman *et al*. Proximity search in databases. In *VLDB*'98.
9. H. Jagadish, A. Mendelzon, and T. Milo. Similarity based queries. In *PODS*, 1995.
10. D. Lewis. *www.research.att.com/ lewis/reuters21578.html*, 1997.
11. R. Subramonian. Defining *diff* as a data mining primitive. In *KDD* 1998.
12. M. J. Zaki *et al*. New algorithms for fast discovery of association rules. In *KDD*, 1997.
13. M. J. Zaki *et al*. Evaluation of Sampling for Data Mining of Association Rules In *RIDE*, 1997.

[4] We emphasize here that the influential attribute detection algorithm was able to automatically identify the most influential attributes causing the similarity in these cases as opposed to requiring supervision [5].