Distance Based Clustering of Association Rules

Alexander Strehl Gunjan K. Gupta Joydeep Ghosh
The University of Texas at Austin
Department of Electrical and Computer Engineering
Austin, TX 78712-1084, U.S.A.

***Abstract*— Association rule mining is one of the most important procedures in data mining. In industry applications, often more than 10,000 rules are discovered. To allow manual insepection and support knowledge discovery the number of rules has to be reduced significantly by techniques such as pruning or grouping. In this paper, we present a new normalized distance metric to group association rules. Based on these distances, an agglomerative clustering algoritm is used to cluster the rules. Also the rules are embedded in a vector space by multi-dimensional scaling and clustered using a self organizing feature map. The results are combined for visualization. We compare various distance measures and illustrate subjective and objective cluster purity on results obtained from real data-sets.**

I. Introduction

A. Motivation

Massive amounts of data are being generated and stored every day in corporate computer database systems. Mining association rules [2] from transactional data is becoming a popular and important knowledge discovery technique [3]. For example, association rules (ARs) of retail data can provide valuable information on customer buying behavior. The number of rules discovered in a real data-set can easily exceed $10,000$. To manage this knowledge, rules have to be pruned and grouped, so that only a reasonable number of rules have to be inspected and analysed. In this paper we propose a new distance metric between two ARs. and propose a new grouping methodology using multi-dimensional scaling (MDS) and self organizing maps (SOMs). The following subsection defines the terms used in this paper.

B. Notation

A relational database r consists of a large set of transactions t. A market-basket (MB) is the set of all the transactions occurring during one shopping session of a customer. Each market-basket consists of one or more items I, which are either present (1) or absent (0). Let us look at the following exemplary association rule (AR)

$$\text{Electric Wiring, Light Bulb} \rightarrow \text{Lighting Fixture } (0.01, 0.7)$$

This association rule expresses that, when a customer buys Electric Wiring and a Light Bulb, there is a 70% chance that he also buys a Lighting Fixture at that time. Moreover, 1% of all market-baskets contain all three items. The above rule can be abbreviated as follows -

$$A, B \rightarrow C \text{ (support, confidence)}$$

Each, A, B and C is an item I and, thus, an element of the set of all possible items R. They can be grouped into three different item-sets. The left-hand-side (LHS) $\{A, B\}$, the right-hand-side (RHS) $\{C\}$, and both sides (BS) together $\{A, B, C\}$. Generally speaking, a rule can be written in the following form.

$$LHS \rightarrow RHS\ (s, c) \tag{1}$$

Where

$$LHS \subseteq R \tag{2}$$

$$RHS \subseteq R \setminus LHS \tag{3}$$

$$BS = LHS \cup RHS \tag{4}$$

The set of all transactions t containing a particular item-set X is denoted with $m(X)$. The support s of a rule is defined as follows -

$$s = \frac{|m(BS)|}{|r|} \geq \sigma \tag{5}$$

where $|r|$ is the number of transactions in the database r. Unlike logic rules, association rules do not have to hold in all cases. The following defines the confidence c of a rule -

$$c = \frac{|m(BS)|}{|m(LHS)|} \geq \gamma \tag{6}$$

In the association rule mining process, both support and confidence, have to exceed a certain threshold in order to be discovered. These thresholds are denoted σ and γ, respectively.

C. Approach

In this paper, we propose a new distance metric to cluster association rules (section II) which improves the metric proposed in [8]. Based on the distance metric, we propose a new agglomerative clustering technique (section III). Moreover, we embed the distances using multi-dimensional scaling and cluster the resulting points into a Euclidean space using a Self Organizing Feature Map (SOM)(section IV). We propose a visualization scheme to compare both techniques by color-coding the SOM results based on the agglomerative clustering results (section IV). Figure 1 depicts the overall process flow-diagram of our proposed system.

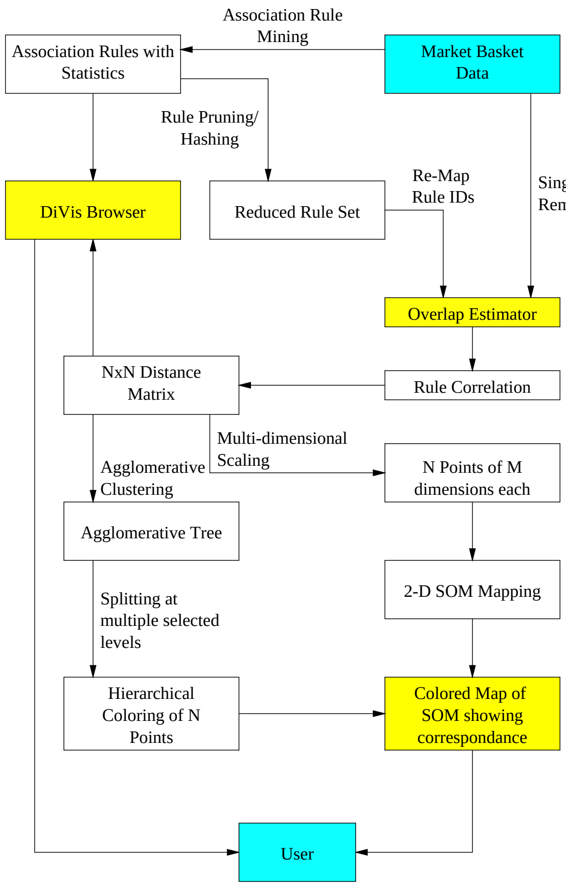

Fig. 1. Flow diagram of distance computation and clustering process.

II. Distance Metrics

To cluster rules we first need a way of quantifying how far two rules are apart. Unfortunately, there is no intuitive distance metric for association rules. A Euclidean distance could be defined on rule features such as support, confidence, lift or the bit-vector representation of BS. These direct features are very limited in capturing the interaction of rules on the data and characterize only a single rule. However, for a deeper analysis of the relationship of two rules, we have to resort to their common origin; the data. We call rule distances that are obtained from the data Indirect Distance Metrics.

An indirect distance is defined as a function of the market-basket sets that support the two considered rules. For the purpose of distance computation, we assign all rules with the same set of supporting market-baskets the same association rule identifier. For example the following six rules are considered equivalent, because all involve the item-set $\{A, B, C\}$ and all have the same set of supporting market-baskets.

$$\begin{array}{l} A \rightarrow B, C \\ A, B \rightarrow C \\ A, C \rightarrow B \\ B \rightarrow A, C \\ B, C \rightarrow A \\ C \rightarrow A, B \end{array}$$

The following three subsections describe three distance metrics for association rules, based on Absolute Market-basket Difference (AMBD), Conditional Market-basket Probability (CMBP), and Conditional Market-basket Log-likelihood (CMBL).

A. Absolute Market-basket Difference Distance

One way to define distance between rules is in terms of the number of market-baskets that they differ in (meaning one rule is supported, but not the other). Based on the number of non-overlapping market-baskets, a distance metric $\mathbf{d}^{AMBD}$ between rule i and rule j can be defined similar to the metric proposed in [8] as given by equation 7.

$$\begin{aligned} d_{i,j}^{AMBD} &= |(m(BS_i) \cup m(BS_j)) \setminus m(BS_i, BS_j)| \quad (7) \\ &= |m(BS_i)| + |m(BS_j)| - 2 \cdot |m(BS_i, BS_j)| \end{aligned}$$

Rules valid for exactly the same baskets have a distance of zero. Rules applying to disjoint sets of baskets have a distance equal to the sum of the numbers of transactions for which each rule is valid. There are several problems with this measure. One such problem is that it grows as the number of market-baskets in the database increases. This can be corrected by normalizing (divide the measure by the size of the database $|r|$). However, the measure is still strongly correlated with support. High support rules will on average tend to have higher distances to everybody else. This is an undesired property. For example, two pairs of rules, both pairs consisting of non-overlapping rules, may have different distances. High support pairs have a higher distance than low support pairs. As an improvement to this metric, we propose a new distance measure based on a conditional probability estimate as described in the following subsection.

B. Conditional Market-basket Probability Distance

Using a probability estimate for distance computation has many advantages. Probabilities are well understood, are intuitive, and a good measure for further processing. The distance $d_{i,j}^{CMBP}$ between to rules i and j is the (estimated) probability that one rule does not hold for a basket, given at least one rule holds for the same baskets (equation 8).

$$\begin{aligned} d_{i,j}^{CMBP} &= P(\overline{BS_i} \vee \overline{BS_j} | BS_i \vee BS_j) \quad (8) \\ &= 1 - \frac{|m(BS_i, BS_j)|}{|m(BS_i)| + |m(BS_j)| - |m(BS_i, BS_j)|} \end{aligned}$$

With this metric, rules having no common MBs are at a distance of 1, and rules valid for an identical set of baskets are at a distance of 0. This metric can also be interpreted as a normalized AMBD. The CMBP is computed as the AMBD divided by the number of baskets in the joint set of each rule's supporting basket set. The CMBP does not suffer from the support correlation problem of AMBD. Let us call a distance interesting if it is neither 0 nor 1. Rule pairs with an interesting distance are called good neighbors. In most real databases, the majority of all rule pairs are not good neighbors. Manual exploration of a rule's good neighbors showed that intuitive relatedness was captured very well by this metric.

For example, rules involving different items but serving equal purposes were found to be close good neighbors. Super-set relationships of the item-sets associated to the rules often lead to very small distances.

C. Conditional Market-basket Log-likelihood Distance

Due to its probabilistic nature, the range of the CMBP distance space is limited between 0 and 1. Natural distances usually range from 0 to infinity. Also, most of the interesting distances are in the open interval between .9 and 1 and the majority of the distances are exactly 1. This property may induce a high embedding dimensionality and, hence, cause problems in the clustering stage. A better behaved distance is given by the log-likelihood distance, which is defined in equation 9.

$$d_{i,j}^{CMBL} = -\log\left(\frac{|m(BS_i, BS_j)|}{|m(BS_i)| + |m(BS_j)| - |m(BS_i, BS_j)|}\right) \tag{9}$$

Non-overlapping rules are at a distance of infinity. Rules with identical support have a distance of 0. This metric monotonically spreads the CMBP range of 0 to 1 onto 0 to infinity. Thus, at any point the distance based ordering or ranking will remain the same as for the CMBP. However, the area of good neighborhood is 'magnified'. Thus, interesting patterns in this region may be easier to identify for the user when visualized. Also, this seems better suitable for the clustering discussed later.

D. Algorithm and Complexity of Distance Computation

Before computing the actual distances, the overlap of all rule pairs on the market-basket data has to be estimated. The overlap count $|m(BS_i, BS_j)|$ can be represented in matrix form $\mathbf{o}$ such that the entry $o_{i,j}$ at row i and column j holds the absolute number of market-baskets that support both rules i and j. The diagonal elements fall out to be the absolute support of the corresponding rule.

We initialize the overlap matrix as a zero matrix. Then, we scan the database one market-basket at a time. For each basket we generate a list of rules that are discovered and supported. In the worst case, this list grows exponentially with the number of items in the market-basket. On an average, the lists' growth is bound by the number of rules discovered in the ARM stage, which are only a very small fraction of all possible rules. The rule list represents all discovered and supported rules in the current market-basket. We increment the support of each rule i on the list by adding 1 to the current value of $o_{i,i}$. Then, we construct all possible rule pairs from the list of discovered and supported rules. Let m denote the number of items in the current market basket. There are $\binom{m}{2}$ possible rule pairs. Each unordered pair corresponds to two symmetric locations $o_{i,j}$ and $o_{j,i}$ in the overlap matrix. We increment both matrix entries by 1 to indicate that a basket where both rules apply has been found. We do so for each pair of different rules on the basket's rule list.

This procedure is repeated for the entire data-set or a sub-sample thereof to obtain the final overlap matrix $\mathbf{o}$. However, depending on the data mining framework, it may not be necessary to compute all of the matrices entries, since some of these frequencies may already be available as the item-set counts from the association rule discovery stage. After the overlap matrix is estimated, the entries of the distance matrix can now be computed using one of the formulas described in the previous subsections.

The average computation time complexity for the computation of d is given in equation 10.

$$O(N \cdot M^2 + K^2) \tag{10}$$

N is the number of transactions $|r|$ in the database r, M is the average market-basket size in number of transactions and K is the number of discovered rules. The entire distance matrix $\mathbf{d}$ is currently computed in one extra databases scan. The AprioriTID algorithm [1] mines association rules in a single database scan by storing transaction ids (TIDs). With a fairly reasonable overhead, it may be possible to compute the distance while mining the association rules, so that no extra database scan is required.

The memory space complexity grows as $O(N+K^2)$, which is quadratic with K. However, since the transactions are usually already stored in a database, N does actually not scale memory requirements. A sparse matrix representation for $\mathbf{d}$ can cut down memory requirements significantly. The archived savings, however, depend strongly on how correlated most rules are. The more uncorrelated rule pairs there are, the higher are the savings in memory.

III. Clustering

The distances described in the previous sections are measures of dissimilarity between pairs of rules. Multi-Dimensional Scaling[6] can be used to convert distance information into an embedded for all the rules into a vector space such that the distance between pairs of rules is reasonably preserved. Each rule can then be represented as a point in an N-dimensional space. Another problem is visualization of clustering results. The number of rules is large and their textual representation very cumbersome. Presenting the rule clusters as points in a space is also not feasible until we know the space of the clusters. SOM provides us with one such pseudo-space but to train the SOM itself we need an input vector for each rule.

Another possibility for obtaining an embedding space for the rules is by defining a binary vector for each rule with one bit per item to describe its presence or absence. But such vectors are very sparse since the number of different items runs into thousands. The approach does not seem very attractive especially from the point of view of training a neural network. Similar problems occur with multi-dimensional scaling although to a much lesser extent. One reason for this to happen is the lack of transitivity in the distance relationship between rules. For example if rule i is close to rule j and rule k is also close to rule j, it does not imply that rule i is close to rule k. But this relationship is true in a vector space.

Hence the number of dimensions obtained by multi-dimensional scaling for a sufficient accuracy of distance representation remains high. But it is still small compared to the bit-vector representation. A hierarchical clustering technique allows the results to be interpreted at different levels of cluster compactness. There are many approaches to clustering but most of them need the coordinates to estimate the cluster center. We propose a new algorithm referred to as Dimensionless Agglomerative Chain Clustering.

A. Chaining

The Chaining algorithm does not use any coordinate system and finds the clusters using the distance measures only. In this algorithm a point is joined to its closest neighbor. This is found by looking at the distance matrix. This process is applied to all the points in the space and at the end of it we get a collection of graphs. All points joined together as a graph end up having the same label. The algorithm returns the labels of all the points. A nice property of the algorithm is that it scales the cluster sizes depending upon the density of the points in the neighborhood. A more dense neighborhood results in a smaller more compact cluster and a more sparse region of the space returns a larger less dense cluster. It can be shown that the resultant clusters are unique and do not depend on the starting point. Chaining is performed as follows:

1) $i = 0$, $labelcount = 0$
2) If i is less than number of points $then$ increment i, $else$ Go to Step 9.
3) Find the closest point P_j to point P_i.
4) If the point P_i and P_j both do not have labels $then$ increment $labelcount$ and assign both points the label $labelcount$ (creates a new cluster with points P_i and P_j as members).
5) If the point P_i has a label and P_j does not have a label $then$ assign P_j the label of P_i (merging of point P_j into the cluster of P_i).
6) If the point P_i does not have a label and P_j has a label $then$ assign P_i the label of P_j (merging of point P_i into the cluster of P_j).
7) If the point P_i and P_j both have labels $then$ assign all points having a label same as P_i the label of point P_j. (This results in merging of cluster of Pi into cluster of P_j).
8) Go to Step 2.
9) Find all cluster labels and change them such that the cluster labels are from 1 to N_c where N_c is the number of unique cluster labels.
10) Return the points and their labels.

B. Agglomerative Chain Clustering

Agglomerative Chain Clustering performs Chaining at multiple levels. At the end of the algorithm we get a tree structure that describes the multiple levels of clustering. It is similar to Single Link Agglomerative Clustering[4] but differs in its bias. The tree returned is shorter and the clusters more uniformly sized. The algorithm works as follows:

1) Perform chaining on all the points as described in previous section and retrieve all the clusters.
2) For each cluster find the cluster center as the centroid by taking the mean coordinates of all the level 1 (original) points in that cluster.
3) Form a new space of N_c point representing the cluster centers.
4) If N_c is greater than 1 $then$ Go to Step 1 $else$ STOP.

At the end of the run each point gets labeled with multiple labels, each successive label describing its position in the tree at that level. All the points merge to form a single cluster at the top level. At the bottom level each point is a cluster in itself. An Agglomerative Chained Tree is shown in Figure 2. In this figure the height of a node is calculated as the average distance of the original points of the cluster from the centroid. This height represents the compactness of the clusters and is useful for extracting clusters of comparable compactness from the tree.

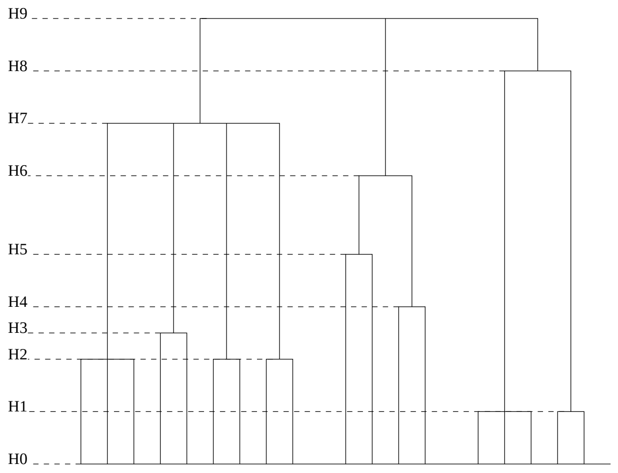

Fig. 2. An example of an Agglomerative Chain Clustering Tree with nine unique splitting points each resulting in a different cluster set. For example splitting at H4 would give the following cluster sets (1,2,3), (4,5), (6,7), (8,9), (10), (11), (12,13), (14,15,16), (17,18). H4 is also the height of nodes 12 and 13. Nodes are numbered from left to right.

C. Dimensionless Agglomerative Chain Clustering

This is the method used for clustering Association Rules in this paper. It is a special case of the Agglomerative Chain Clustering and allows us to cluster rules together even in the absence of dimensional information. As explained in Section III-A the first level chaining performs the clustering without any information of the location of the points. To repeat the chaining at the next level, we only need the distance between clusters (centers) and not the coordinates of the centers themselves. These distances can be estimated using many different methods. The method is used in this paper is a variation of the Lance and Williams Flexible Method[4]. According to their method, the distance between a group k and a group (ij) formed by the fusion of groups i and j satisfies the recurrence formula for the distance defined as follows:

$$d_{k(i,j)} = \alpha_i d_{ki} + \alpha_j d_{kj} + \beta_{dij} + \gamma|d_{ki} - d_{kj}| \quad (11)$$

where d_{ij} is the distance between the groups i and j and α, β and γ are parameters whose values depend on the definition of the center of clusters. By allowing β to vary, clustering schemes with various characteristics can be obtained; Lance and Williams suggest that probably the best value to assume for β is some small negative value, and in their example they use the value 0.25. For centroid the parameters α, β and γ take the following values:

$$\alpha_i = \frac{n_i}{n_i + n_j}; \alpha_j = \frac{n_j}{n_i + n_j}; \beta = -\alpha_i \alpha_j; \gamma = 0 \tag{12}$$

For three points, the formula becomes:

$$d_{k(ji)} = 0.5(d_{ki} + d_{kj}) - 0.25 d_{(ij)} \tag{13}$$

where $d_{k(ji)}$ represents the distance of point k from the centroid of cluster ij. As we can see it is equal to the average distance minus one-fourth the distance between i and j. This distance measure is suitable for the single link clustering algorithm described by the author in [4].

In our clustering algorithm more than two points merge at one level. Hence the formula had to be modified to make it applicable. Using it, we can estimate distances between clusters at level $l+1$ using centroid distances between points at level l. The distance between clusters a and b is given by the formula:

$$C_{ab} = \frac{1}{n_a n_b} \sum_{i=1}^{n_a} \sum_{j=1}^{n_b} d_{ij} \tag{14}$$

$$A_a = \frac{1}{n_a^2} \sum_{i=1}^{n_a} \sum_{j=1}^{n_b} d_{ij} \tag{15}$$

$$A_b = \frac{1}{n_b^2} \sum_{i=1}^{n_a} \sum_{j=1}^{n_b} d_{ij} \tag{16}$$

$$d_{ab} = C_{ab} - 0.5(A_a + A_b) \tag{17}$$

where n_a represents the number of (original) points in cluster a and n_b represents the number of (original) points in cluster b. For the height of a node a in a tree the formula used is the mean distance of all the points in the cluster represented by the node as shown below.

$$H_a = \frac{2}{n_a(n_a - 1)} \sum_{i=1}^{n_a} \sum_{j=1} d_{ij} \tag{18}$$

The points in the cluster a are represented by the leaf nodes of the agglomerative tree that have node a as an ancestor. n_a is the number of points in cluster a.

We examined the quality of dimensionless clustering by considering only the distance information between data points in different dimensional Euclidean spaces. For a very high dimensional space the error was small and since the Association Rule space is inherently high dimensional, the formulas for cluster width and distance estimation work well. The clusters obtained during simulations using the dimensionless approximation are identical to dimension based clustering for up to 60 points for a 2-D space and even more for higher dimension space. At 1000 points and 40 dimensions the number of points grouped differently in the clustering tree is less than 1% on an average.

D. *Comparison of Chain Clustering and Single Link Clustering*

Single Link Clustering [4] algorithm is a popular Agglomerative Clustering algorithm. Both algorithms have a complexity of $O(N^3)$ although the average case of chaining is faster. Also when the Agglomerative Chain Clustering clusters points that are in a sparse area of a common space, it dynamically scales the clusters in that region.

IV. SOM Clustering & Visualization

A. *Multi-Dimensional Scaling*

The scalar distance between the rules cannot be used as an input to a SOM directly. Hence Multi-Dimensional Scaling is performed using Singular Value Decomposition. The data is normalized to zero mean distribution and the suitable value for the embedding dimension is obtained by monitoring the Stress Factor[6]. Given the Matrix M as the original distance matrix and M' as the corresponding matrix in the projected space, the Stress between M and M' is given by:

$$Stress = \frac{\sum_{k=1}^{N} \sum_{i=1}^{k-1} (d_{ik} - d'_{ik})^2}{\sum_{k=1}^{N} \sum_{i=1}^{k-1} (d_{ik})^2} \tag{19}$$

Analysis on simulated data suggests that the Multi-Dimensional Scaling works well for an inherently N-dimensional Euclidean space; the Stress drops suddenly to a very small value for all test dimensions L greater than N-1. The experiment was repeated for various simulated data sets. A Stress threshold with a shortened binary search gives a very close estimate of the correct number of dimensions in 2-4 trials. For a search range of 1 to N, the Shortened Binary Search examines stress value at $L = \frac{N}{2}, \frac{3N}{4}, \frac{7N}{8}$, .. until the Stress is less than threshold and then terminates. Since the Stress is a decreasing function of L, this is possible. Figure 2 shows the results on actual data from Knowledge Discovery One. For 1000 rules the cutoff is reached with Shortened Binary Search at L=750 with 2.3 % Stress which is good enough for the 5 % threshold used. For the same data-set a full binary search requires many more iterations.

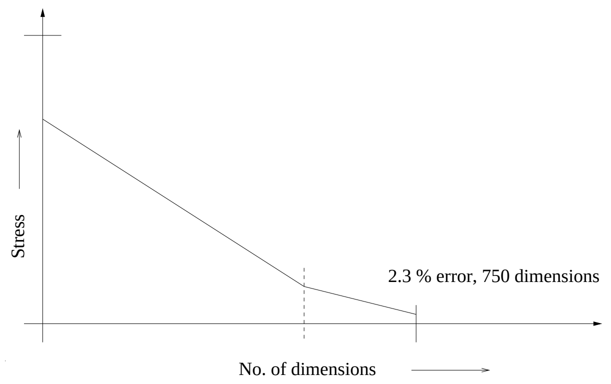

Fig. 3. Multi-Dimensional Scaling : Finding the right number of dimensions using a shortened binary search. The example shown enough finds the number number of dimensions as 750 in 2 iterations for a 1000x1000 distance matrix. The Stress is 2.3%. The threshold used is 5% Stress.

B. Training of the SOM

The embedded space obtained from Multi-Dimensional Scaling can now be used with any clustering algorithm that needs a vector input. In particular, mapping the input space to a 2-D SOM output space seemed to be very suitable since it provides an easy visualization of the clustering. But how do we verify results from SOM given the abstract nature of 'distance' between rules ? We developed a novel technique based on defining a hierarchical color spectrum over the Agglomerative Clustering Tree. Thus the more similar two clusters are in color, the more closely they appear together in the Agglomerative Clustering tree. This allows us to evaluate our SOM results by coloring the SOM with the colors from the Agglomerative Clustering Tree hierarchy.

V. Results

A. Distance

The first test data-set consists of 172,000 cash register transactions of a home improvement store. From this data 2831 frequent item sets, 4782 association rules and 1311 hashed association rules are extracted. The 1311-by-1311 distance matrix images for AMBD, CMBP and CMBL (top to bottom) are shown in Figure 4. Black always corresponds to a distance of 0 and white to 711 (AMBD, top), 1 (CMBP, middle) or 6.5294 (CMBL, bottom).

The vertical and horizontal lines in the AMBD image illustrate the strong support dependencies. High support rules tend to have higher distances to every other rule, so their entire row (and due to symmetry, their entire column) appears as a bright line.

The new CMBP meets the intuitive expectations of a distance metric much better. Interesting pairs contrast very well as dark spots in the image. The median distance is 1 meaning that most rule pairs are unrelated. The mean distance is 0.9943. 92.97% of the distances are 1. Most rules have between 1% and 20% good neighbors.

In visualizing the CMBL distance, the good neighbors appear enhanced since the most densely populated distance space between .9 and 1 gets spread.

B. Clustering and SOM

Agglomerative Clustering was performed on a rules distance matrix of size 1,311x1,311. The number of different levels available for splitting the tree obtained is 289. The split that is used for defining the colors of the SOM is such that it results in 19 clusters at 208th split level and 715 clusters at 210th split level. Each of the 1311 rules is colored using a color formed by combining the values of the two labels at the two different level so that the colors reflect the closeness of two points. These colors were then used to plot the results of the SOM. The tree generation using Dimensionless Agglomerative Chain Clustering (section IV-A) is fast. It requires less than half the time compared to Multi-Dimensional Scaling on the same distance matrix with 5% Stress threshold. The web-site http://www.ece.utexas.edu/ gunjan/aclu/ has SOM results for various number of epochs. Some of the results are included with this paper as colored printouts.

Clusters of relatively pure color clearly show the correlation between the clusters discovered by Dimensionless Agglomerative Chain Clustering algorithm and the SOM. Given that SOM is mapping 715 clusters onto a 10x10 grid, we expect on an average of 7 clusters to fall onto one point. Most of the overlaps should be with other clusters that are close to the given clusters. This should result in a color localization on the SOM.

A SOM with only 40 input dimensions needs a much smaller training time than the full 965 dimensions. There is considerable representational error if only 40 dimensions are used but surprisingly the SOM still seems to find many good clusters.

Since there is no ground truth for verification of the results, the visualization using SOM allows us to only get a good overall idea of the clusters. A good overlap between SOM and Agglomerative Clustering might imply that the clusters are more reliable. It also allows the user to inspect such clusters first. But the only way to see if the clusters are good is by printing out the text form of the Association Rules. As in the examples shown below, the rules do appear to be correlated as opposed to randomly picked rules.

- Randomly picked rules.
  1) ARID 738
     - LIGHTING-FIXTURES-FLUORES.-UTILITY LIGHT-BULBS-INCANDESCENT
  2) ARID 616
     - 17434 PLUMBING-CEMENTS-AND-PRIMERS
     - 17442 PLUMBING-P/F—PVC-PRES.-FTGS.-SCH40
  3) ARID 140
     - 12048 LIVE-NURSERY-ANNUALS
     - 12060 LAWN-&-GARDEN—MULCH,-SOILS,-PEAT

  ..

- A subset of a cluster from Dimensionless Agglomerative Chain Clustering -
  1) Rule 278932 (278932)
     - 16846,PAINT BRUSHES (new)
     - 16926,PAINT-INTERIOR-ONE & ONLY (new)
     - 16844,PAINTING DROPCLOTHS (new)
  2) Rule 278773 (278773)
     - 16846,PAINT BRUSHES (new)
     - 16871,PAINTING ACC. - SHURLINE (new)
     - 16730,PAINT-MISC SUNDRY ITEMS (new)
  3) Rule 277258 (277258)
     - 16840,PAINTING ACCESSORY ITEMS (new)
     - 16926,PAINT-INTERIOR-ONE & ONLY (new)
     - 16844,PAINTING DROPCLOTHS (new)
  4) Rule 277269 (277269)
     - 16871,PAINTING ACC. - SHURLINE (new)
     - 16926,PAINT-INTERIOR-ONE & ONLY (new)
     - 16844,PAINTING DROPCLOTHS (new)
  5) Rule 277433 (277433)
     - 16871,PAINTING ACC. - SHURLINE (new)
     - 16730,PAINT-MISC SUNDRY ITEMS (new)
  6) Rule 277435 (277435)

  ... and so on.

- A subset of a cluster from SOM
  1) Rule 278971 (278971)
     - 17160,LIGHT BULBS-INCANDESCENT (new)
     - 16720,SC JOHNSON (new)
  2) Rule 278926 (278926)
     - 16794,ETTMORE (new)
     - 16723,RECKITT & COLMAN (new)
  3) Rule 278923 (278923)
     - 16794,ETTMORE
     - 16720,SC JOHNSON
  4) Rule 278765 (278765)
     - 16790,QUICKIE (new)
     - 16723,RECKITT & COLMAN
  5) Rule 278763 (278763)
     - 16723,RECKITT & COLMAN
     - 16720,SC JOHNSON
  6) Rule 278753 (278753)
     - 12360,BAGS-TRASH & LEAF (new)
     - 16790,QUICKIE
  7) Rule 277315 (277315)

- 17160,LIGHT BULBS-INCANDESCENT
- 16790,QUICKIE

8) Rule 277425 (277425)
   - 16794,ETTMORE
   - 16790,QUICKIE
9) Rule 277563 (277563)
   - 17160,LIGHT BULBS-INCANDESCENT
   - 16723,RECKITT & COLMAN
10) Rule 277728 (277728)
   - 16790,QUICKIE
   - 16720,SC JOHNSON

.... and so on.

VI. Conclusions and Future Work

A key reason for clustering rules is to obtain more concise and abstract descriptions of the data. We plan on merging rules of the same cluster into joined meta-rules. However, this is not a trivial problem. We are currently investigating the use of meta-data (such as product hierarchies) to support merging decisions.

More powerful agglomerative clustering techniques that have other height definitions and splitting properties may yield better clusters. Trying out other distance measures for cluster centers is required for comparing results. Another idea is to modify the SOM to allow training on dimensionless distance data.

Another alternative worth exploring is to represent data at multiple levels of product hierarchy before extracting, clustering and merging of association rules. This may provide more abstract descriptions of the data's association rules that better capture customer buying behavior. By keeping track of customer behavior, a home-improvement store, for example, could make inferences on what project a specific customer is working on and provide exactly the right service to satisfy that customer's needs.

Acknowledgements

We would like to thank Knowledge Discovery One for providing us with real world data-sets and especially Mark Davis for his support and suggestions.

References

[1] R. Agrawal and R. Srikant. Fast algorithm for mining association rules in large databases. In *Proceedings 20th International Conference on Very Large Data Bases*, pages 478–499, September 1994.

[2] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *SIGMOD-93*, pages 207–216, May 1993.

[3] Ming-Syan Chen, Jiawei Han, and Philip S. Yu. Data mining: An overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):866–883, December 1996.

[4] Brian Everitt. *Cluster Analysis, 2nd Edition*, chapter 3. Halsted Press, 1980.

[5] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proceedings 21th International Conference on Very Large Data Bases*, pages 420–431, September 1995.

[6] Richard A. Johnson and Dean W. Wichern. *Applied Multivariate Statistical Analysis*, chapter 12. Prentice Hall, 1982.

[7] Jong Soo Park, Ming-Syan Chen, and Philip S. Yu. Using a hash-based method with transaction trimming for mining association rules. *IEEE Transactions on Knowledge and Data Engineering*, 9(5):813–825, September/October 1997.

[8] H. Toivonen, M. Klemettinen, P. Ronkainen, and H. Mannila. Pruning and grouping discovered association rules. In *MLnet Workshop on Statistics, Machine Learning and Discovery in Databases*, pages 47–52, April 1995.

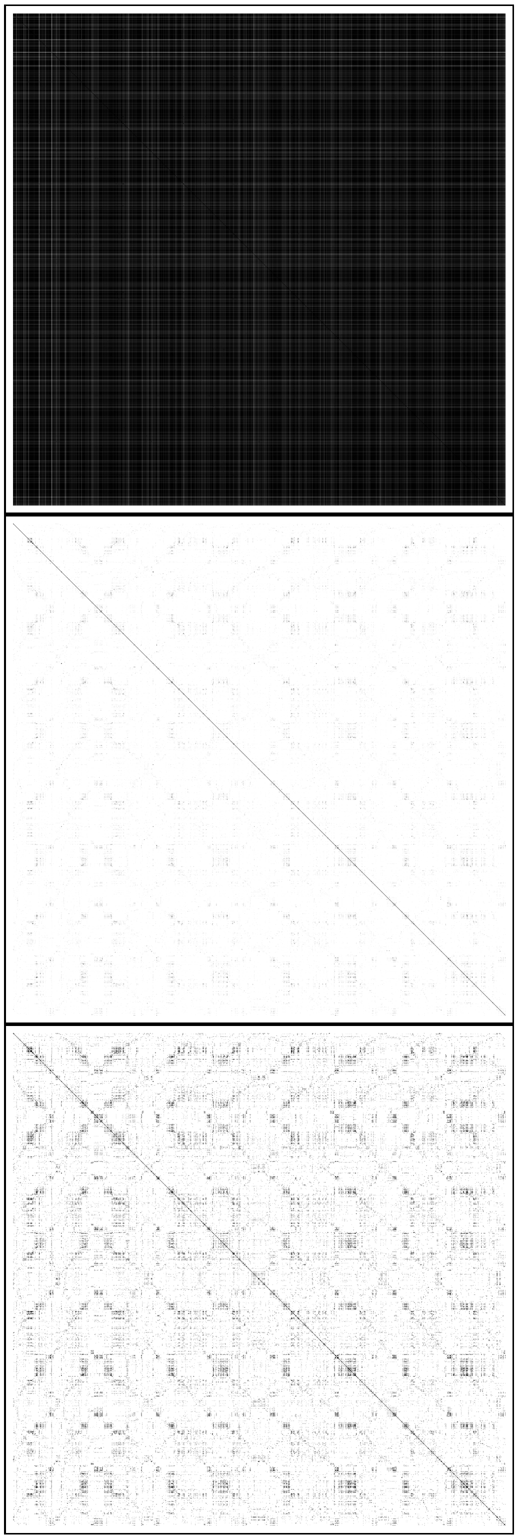
Fig. 4. Visualization of different distance metrics for the same dataset. Image pixel brightness increases with rule distance. Distances based on AMBD (top), CMBD (middle), and CMBL (bottom) are shown for the 1311 rules of a home-improvement retail store.