Context-Based Similarity Measures for Categorical Databases

Gautam Das[1] and Heikki Mannila[2]

[1] Microsoft Research
gautamd@microsoft.com
[2] Nokia Research
Heikki.Mannila@nokia.com

Abstract. Similarity between complex data objects is one of the central notions in data mining. We propose certain similarity (or distance) measures between various components of a 0/1 relation. We define measures between attributes, between rows, and between subrelations of the database. They find important applications in clustering, classification, and several other data mining processes. Our measures are based on the contexts of individual components. For example, two products (i.e., attributes) are deemed similar if their respective sets of customers (i.e., subrelations) are similar. This reveals more subtle relationships between components, something that is usually missing in simpler measures. Our problem of finding distance measures can be formulated as a system of nonlinear equations. We present an iterative algorithm which, when seeded with random initial values, converges quickly to stable distances in practice (typically requiring less than five iterations). The algorithm requires only one database scan. Results on artificial and real data show that our method is efficient, and produces results with intuitive appeal.

1 Introduction

Similarity between complex data objects is a crucial notion in data mining and information retrieval. In order to look for patterns or regularities in a database, it is often necessary to be able to quantify how far from each other two objects in the database are. Once we have a natural notion for similarity between objects in a database, we can for example use distance-based clustering or nearest neighbor techniques to search for interesting information from the data set. Recently, there has been considerable interest in defining intuitive and easily computable measures of similarity between complex objects and in using abstract similarity notions in querying databases [1, 2, 10, 14, 17, 19, 22, 4, 12, 15].

Ideally, the similarity notion is defined by the user, who understands the domain concepts well and is able to explicate the notions needed for similarity computations. However, in many applications the domain expertise is not available. The users do not understand the interconnections between objects well enough to formulate exact definitions of similarity or distance. In such cases it is quite useful to have (semi)automatic methods for finding out the similarity (or distance) between objects.

D.A. Zighed, J. Komorowski, and J. Zytkow (Eds.): PKDD 2000, LNAI 1910, pp. 201−210, 2000.

In this paper we focus on defining similarity between various components of *categorical databases*. Such databases typically do not contain numeric data. Instead, the domains of the attributes are small, unordered sets of values. An important example is a *market basket database* which is typically a supermarket's record of customer purchases. Conceptually this is a binary table where columns (or attributes) represent products, and rows represent customers. Such databases also frequently occur in domains outside retail (for example, in this paper we consider a TV viewership dataset, where the columns are TV shows and the rows are viewers). We consider the problem of defining distance measures between various components of such databases, such as between attributes, between rows, and between subrelations. Since the data is nonnumeric, simple distance measures such as Euclidean distances are inappropriate in this context.

We develop measures that are *context-based*, i.e. similarity between components is determined by examining the contexts in which the components appear. For example, two products (i.e. attributes) are deemed similar if their respective sets of customers (i.e. subrelations) are similar. This reveals more subtle relationships between components, something that is usually missing in simpler, context insensitive measures. A highlight of our methods is that they are based on plausible probabilistic arguments: data components are assumed to be samples of underlying probability distributions, and computing the similarity between two components is reduced to computing the similarity between two respective probability distributions. While other context-based approaches to similarity and clustering problems have been investigated by other researchers (see discussion at the end of this section), our proposal is particularly appropriate for the kind of applications we consider.

In the rest of the paper we describe our results only for market basket databases, i.e. binary tables. Our results can be extended to general categorical databases by mapping them into binary relations; we omit details from this version of the paper. In what follows, we use the words *attribute*, *column* and *product* interchangeably; likewise we also use *row* and *customer* interchangeably.

Consider the problem of defining (dis)similarities between attributes. This has several applications, such as in forming product hierarchies or clusters of attributes [13, 20]. Typically, one assumes that the hierarchy is given by a domain expert. However, in the absence of such knowledge, we could produce a product hierarchy that is derived from the data. This can be done through standard similarity measures such as Euclidean distance, correlations, etc. However, if we use them in a typical supermarket scenario, we might conclude that two products such as Coke and Pepsi are quite dissimilar because they do not have many common customers, whereas in reality both are soft drinks and should thus be more similar than say, Pepsi and Mustard. Secondly, consider a high volume product such as Coke, and a relatively low volume product such as RC Cola. Correlations or Euclidean distances would not detect much similarity between the two. We propose the following measure: two products are similar if the *buying patterns* of the customers of each are similar. Thus Coke and Pepsi can be deemed similar attributes, if the buying behavior of buyers of Coke and buyers

of Pepsi are similar. But buyers of Coke (resp. Pepsi) are simply subrelations of the database. Thus we relate similarity between attributes to similarity between certain subrelations.

Consider the problem of defining similarities between rows. This is affected by the fact that not all attributes are equidistant from each other (for example, a Coke customer is closer to a Pepsi customer than to a Mustard customer). In other words, our row similarity measure depends on our attribute similarity measure. So we see that various different similarity notions are interdependent on one another: row similarity depends on attribute similarity, which depends on subrelation similarity, which in turn depends on row similarity (since a subrelation is basically a set of rows). The main result of this paper is that the problem of simultaneously solving for all distance measures is formulated as a system of nonlinear equations. We present ICD (*Iterated Contextual Distances*), an iterative algorithm which, when seeded with random initial values, converges quickly to stable distances in practice (typically requiring less than five iterations). The algorithm requires only one database scan.

We complete this section with a discussion of related research. Recently, several iterative procedures (quite different from ours) have been developed for other problems, such as document retrieval systems [5, 15], extraction of information in hyperlinked environments [9, 16], and clustering of categorical data [8]. Much of the emphasis in document retrieval is on grouping/clustering documents based on co-occurances of keywords. The problem of clustering values of a categorical database has attracted much attention recently. A variety of approaches have been developed, such as dynamical systems formulations, probabilistic mixture models, combinatorial methods, etc. [8, 3, 12, 7, 11].

However clustering only gives an indirect sense of inter-attribute distances, and in some applications it is important to have the direct distance measure. It is not clear whether the above clustering methods will easily extend to defining and computing the kind of similarities that we desire (such as defining product similarity based on buying patterns of customers, etc.). Attribute distances of a binary database has been investigated in [12, 4]. The idea of using contexts to define attribute distances has been used in [4], but there the distance between subrelations is defined by looking at the marginal distributions for certain *probe* attributes. Also, the paper did not take into account issues such as the dependencies between attribute distances and row distances.

The rest of the paper is organized as follows. We first describe our Iterated Contextual Distances algorithm. We then describe a variety of experiments on artificial and real data. We conclude with some open problems.

2 The Iterated Contextual Distances (ICD) Algorithm

Let r be a 0/1 relation containing n rows, over the set of attributes R where $|R| = m$. As mentioned earlier, given distances between attributes we will define distances between rows, given distances between rows we will define distances between subrelations, and, given distances between subrelations we will define

distances between attributes. This way of defining distances between attributes is circular, and therefore we use an iterative approach. The basic idea is to initially start with an arbitrary distance function d_0 between attributes of R and use that to derive a vector representation for the rows. From this we go to a vector representation for subrelations. This representation gives us a distance metric Δ_0 between subrelations of r. Then the value of $\Delta_0(r_A, r_B)$ is used to get a new distance value $d_1(A, B)$ for attributes. A few iterations of these steps quickly produces a stable set of distances between attributes. We denote the resulting distance function by d^* and call it *the iterated contextual distance* between attributes. From these attribute distances, we can easily compute other stable distances such as row distances and subrelation distances.

For attributes A and B, let $r_A = \{t \in r \mid t[A] = 1\}$ and $r_B = \{t \in r \mid t[B] = 1\}$ be two corresponding subrelations. We shall relate the distance between A and B to the distance between r_A and r_B. Defining distance between attributes by distance between relations might seem a step backwards; we want to define distance between two objects of size $n \times 1$ by reducing this to distance between objects of dimensions $n_A \times m$ and $n_B \times m$, where $n_A = |r_A|$ and $n_B = |r_B|$. However, we will see later that we can rely on some well-established probabilistic notions for defining distance between subrelations.

2.1 From Attribute Distances to Row Representations

Assume we have a distance function d between attributes. We will use this to define distances between rows.

A row in r can also be viewed as an m-dimensional vector with values from $\{0, 1\}$. In what follows, we will often speak of real-valued rows over R as vectors over R. We map each row $t \in r$ to a vector $f(t)$ over R as follows. We build the vector $f(t)$ for row t by combining vectors g_A for each $A \in t$ (i.e., $A \in R$ such that $t[A] = 1$).

For each attribute $A \in R$ define the function g_A from R to $[0, 1]$ by

$$g_A(B) = \frac{K(d(A, B))}{\sum_{C \in R} K(d(A, C))}$$

where K is a *kernel smoothing* function. We only require that K is monotonically decreasing (for example, $K(x) = 1/(1 + x)$).

Note that $\sum_{B \in R} g_A(B) = 1$. It is useful to think of g_A as a probability distribution that represents the amount of *equivalence* of other products with product A. Thus, if a customer bought A, then $g_A(B)$ represents the probability that the customer would have been satisfied buying B instead. If products A and B are completely similar, i.e., $d(A, B) = 0$, then $g_A(A) = g_A(B)$.

Let $t \in r$, and let $A_1, \ldots, A_k$ be the attributes for which t has value 1. Define a vector $f(t)$ where

$$f(t)(C) = c(g_{A_1}(C), \ldots, g_{A_k}(C))$$

where c is a combination function defined as

$$c(g_1, \ldots, g_k) = 1 - \prod_{i=1}^{k}(1 - g_i),$$

i.e., we have

$$f(t)(C) = 1 - \prod_{i=1}^{k}(1 - g_{A_i}(C))$$

corresponding to a disjunction of the different pieces of evidence $g_{A_i}(C)$ for the presence of C.

Using this mapping f, we define the row distance $d(t, u)$ between rows t and u as the L_1 distance between the two vectors, i.e. $L_1(f(t), f(u))$.

Remarks: There are several possible choices for the kernel smoothing function, $K(x)$. If we run the ICD algorithm with different kernel functions, we get different final distances. However, the actual choice of a does not seem to affect the relative *ranks* of the distances by much.

2.2 Distances between Subrelations

Consider two subrelations r_A and r_B of r (say the Coke customers and the Pepsi customers). We would like to define a notion of distance between r_A and r_B. We construct the following sets of vectors for r_A and r_B: $V(r_A) = \{f(u)|u \in r_A\}$ and $V(r_B) = \{f(v)|v \in r_B\}$. Then the task is reduced to defining the distance between two sets of vectors in m-space.

We could view each relation as a point set and use one of a variety of combinatorial methods for defining the distance between them, such as Hausdorff distance, closest pair, etc. [6]. However, we choose to view the subrelations as samples of corresponding underlying probability distributions (e.g. the distributions of Coke and Pepsi customers), and reduce the problem to computing distance between the two distributions. A comprehensive way of doing this would be to fit distributions (such as gaussians) to each sample and then compute the Kulback-Leibler distance between the two. In the interest of simplicity, we simply choose to compute the distance between the two corresponding centroids c_A and c_B of the sets $V(r_A)$ and $V(r_B)$. Thus,

$$\Delta(r_A, r_B) = L_1(c_A, c_B)$$

2.3 Details of the ICD Algorithm

In detail, the process of finding the attribute distances is as follows. Let r be a 0/1 relation over attributes R.

1. Initialize with random seed: Set $d(A, A) = 0$ for all $A \in R$, and let $d(A, B) = d(B, A) = rand()$ for all $A, B \in R$ with $A \neq B$.

2. Normalize distances: Multiply all distances $d(A, B)$ by a constant c so that the average pairwise distance between attributes is 1.
3. Compute attribute vectors: For each $A \in R$, compute the vector $g_A : R \to [0, 1]$ by

$$g_A(B) = \frac{K(d(A, B))}{\sum_{C \in R} K(d(A, C))}.$$

4. Compute row vectors: For each $t \in r$, let $A_1, \ldots, A_k$ be the attributes for which t has value 1. Define a vector $f(t)$ as

$$f(t)(C) = c(g_{A_1}(C), \ldots, g_{A_k}(C)),$$

where $c(g_1, \ldots, g_k) = 1 - \prod_{i=1}^{k}(1 - g_i)$.
5. Form subrelation centers: For each $A \in R$, let c_A be the vector on R defined as the average of the vectors $f(t)$, where $t(A) = 1$.
6. Compute distances between subrelation centers: For each pair of attributes $A, B \in R$, let

$$\Delta(r_A, r_B) = L_1(c_A, c_B).$$

7. Iterate: for each pair of attributes $A, B \in R$, let $d(A, B) = \Delta(r_A, r_B)$. If the method has converged, stop; now the distance function d is the iterated contextual distance function d^*. Otherwise, go to step 2.

2.4 Analysis of the ICD Method

It can be shown that the distances produced by the ICD algorithm satisfy the triangle inequality. We omit details from this version of the paper.

The ICD algorithm seems to always converge very quickly in practice, typically not requiring more than five iterations. A theoretical analysis of the convergence seems quite difficult, as the ICD algorithm is essentially trying to compute the fixed points of a nonlinear dynamical system. In practice, different starting points sometimes (but rarely) lead to different fixed points, and when that occurs, we have to select the solution(s) that satisfies our requirements best. Our approach is to select the solutions with large variances in attribute distances, the rationale being that solutions which "spread" attributes apart are more interesting.

A naive implementation requires I database scans, where I is the number of iterations required for convergence (usually less than five). A much better implementation which requires only one database scan is possible for sparse relations. Here the first scan can be used to prepare certain main memory data structures (such as the distance matrix, and for each centroid the formula which determines how it is updated after every iteration). The remaining iterations can proceed without having to access the database.

3 Experiments

We experimented with three types of data: (a) small illustrative examples, (b) TV shows viewing data, and (c) data on pages visited at the MSN website.

3.1 An Illustrative Example

Consider the following relation:

	A	B	C	D	E	F	G
t_1	1	0	0	0	0	0	0
t_2	1	1	0	0	0	0	0
t_3	1	0	1	0	0	0	0
t_4	0	1	1	0	0	0	0
t_5	0	1	0	0	0	0	0
t_6	1	0	0	1	0	0	0
t_7	0	1	0	0	1	0	0
t_8	0	0	0	0	0	1	0
t_9	0	0	0	0	0	0	1

If we run the ICD algorithm (with kernel smoothing function $K(x) = 1/(1+x)$), it converges quickly to the following attribute distances:

	A	B	C	D	E	F	G
A	0.000	0.032	0.338	0.338	0.338	2.065	2.065
B	0.032	0.000	0.338	0.338	0.338	2.065	2.065
C	0.338	0.338	0.000	0.058	0.058	2.322	2.322
D	0.338	0.338	0.058	0.000	0.076	2.320	2.320
E	0.338	0.338	0.058	0.076	0.000	2.320	0.320
F	2.065	2.065	2.322	2.320	2.320	0.000	0.063
G	2.065	2.065	2.322	2.320	2.320	0.063	0.000

Upon examining these distances (especially close to the main diagonal), we notice that there are essentially three clusters of attributes: A and B are similar to each other, C, D, and E are similar to each other, and F and G are similar to each other.

We give some intuitive reasoning for the above. Let us try to determine the attribute distance between A and B by examining the buying behavior of their customers. The customers buying A buy the same proportion of C as the customers buying B. Also, the proportion of D bought by A's customers is the same as the proportion of E bought by B's customers. If we knew that D and E are themselves similar, then we can conclude that the customers of A and B are very similar, and thus A and B are themselves very similar. But how do we know anything about the similarity between D and E? If we compare the customers of D with the customers of E, we see that they buy A and B respectively. Thus by a circular argument, we see that A and B must be similar, and D and E also must be similar.

We next consider C. The customers buying C buy the same proportion of A (or B) as the customers buying D. Thus C is similar to D (and thus to E).

As for F and G, since their customers have nothing in common with the others, they are far from the other attributes.

3.2 TV Shows Data

We experimented using data about which TV shows people watched. The data set contained 2272 observations (rows) over about 15 shows (columns)[3] Our objective was to compute distances between TV shows based on viewership patterns.

If we examine the ICD attribute distances for the TV shows in sorted order, they make intuitive sense. For example, the most distant/similar pairs are shown in Figure 1.

Most distant pairs	
AMER.FUNNIEST-HM VIDEOS	FRIENDS
ABC WORLD NEWS TONIGHT	FRIENDS
FRIENDS	CBS EVE NEWS-RATHER/CHUN
MAD ABOUT YOU-THU.	CBS EVE NEWS-RATHER/CHUN
Most similar pairs	
FRIENDS	MAD ABOUT YOU-THU
ABC WORLD NEWS TONIGHT	GOOD MORNING, AMERICA
AMER.FUNNIEST-HM VIDEOS	NBC NIGHTLY NEWS
MURDER, SHE WROTE	60 MINUTES

Fig. 1. Distance between shows

One interesting result is that ABC NEWS:NIGHTLINE is close to both NBC NIGHTLY NEWS and CBS EVE NEWS-RATHER/CHUN. Also, ABC WORLD NEWS TONIGHT and GOOD MORNING, AMERICA are close to each other, but tend to be far from most of the other shows. These relationships were not discovered by other methods, such as by computing correlations between the respective shows.

During the experiments, we tried out the following family of kernel smoothing functions: $K(x) = 1/(1 + ax)$, for $a > 0$. The precise value of a did not seem to affect the relative ranks of the attribute distances by much. Also, the ICD algorithm seems to converge to fixed points very quickly from random starting points (typically within five iterations).

[3] The original data had about 130 shows, but it was pruned to contain only those shows that were watched by at least 300 persons.

3.3 MSN Data

We have started an extensive examination of data on the pages visited at the MSN website. The data has been collected over a six months period, and sampled to extract information about the most frequent 5000 visitors. Since MSN is a vast site, we focus on the activities at the top of the hierarchy, i.e. at the top 50 (in number of hits) DNSids. After some aggregation, we prepared a 50×5000 binary table (in sparse form) where the columns are the DNSid's and the rows are visitors. An entry of 1 indicates that the visitor has visited that DNSid during those six months. Our objective is to compute similarities between DNSids based on similarity between their respective visitor sets.

Due to lack of space, the details of the experiments will appear in the full version of this paper. We only mention here that the initial results appear encouraging. As an example, the pair of DNSids *women.msn.com* and *underwire.msn.com* are found similar by our method, but do not appear similar if we simply compute correlations.

4 Open problems

How reasonable and natural are the resulting distance measures? Our preliminary investigations with real data seem to indicate that the ICD method has intuitive appeal. We are carefully investigating if some of the other context-senstive methods can be unified with ICD to produce even better results in our application domains.

Even though it seems to work well in practice, the ICD method needs further theoretical study. The criteria (if any) for convergence, estimates of the number of fixed points, etc., need to be investigated.

References

1. R. Agrawal, C. Faloutsos and A. Swami. Efficient similarity search in sequence databases. In *Proc. of the 4th Intl. Conf. on Foundations of Data Organization and Algorithms (FODO'93)*, 1993.
2. R. Agrawal, K.-I. Lin, H. S. Sawhney and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Proc. of the 21st Intl. Conf. on Very Large Data Bases (VLDB)*, 1995, pp 490–501.
3. P. Cheeseman and J. Stutz. Bayesian classification (Autoclass): theory and results. In *Advances in Knowledge Discovery and Data Mining*, MIT Press, 1996, pp. 153–180.
4. G. Das, H. Mannila, and P. Ronkainen. Similarity of attributes by external probes. In *Proc. of the 4th Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, 1998, pp 23–29.
5. S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the Society of Information Science*, 41(6), 1990, 391–407.
6. T. Eiter and H. Mannila. Distance measures for point sets and their computation. *Acta Informatica*, 34(2), 1997, pp. 109–133.

7. Venkatesh Ganti, J. E. Gehrke, and Raghu Ramakrishnan. CACTUS–clustering categorical data Using summaries. In *Proc. of the 5th Intl. Conf on Knowledge Discovery and Data Mining (KDD)*, 1999.
8. D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: an approach based on dynamical systems. In *Proc. of VLDB*, 1998, pp. 311–322.
9. D. Gibson, J. Kleinberg, and P. Raghavan. Inferring Web communities from link topology. In *Proc. 9th ACM Conference on Hypertext and Hypermedia*, 1998.
10. D. Q. Goldin and P.Kanellakis. On similarity queries for time-series data: Constraint Specification and Implementation. In *Intl. Conf. on Principles and Practices of Constraint Programming*, 1995.
11. S. Guha, R. Rastogi and K. Shim. ROCK: A Robust Clustering Algorithm for Categorical Attributes. In *Proc. of ICDE*, 1999, pp. 512–521.
12. E.-H. Han, G. Karypis, V. Kumar and B. Mobasher. Clustering Based On Association Rule Hypergraphs. In *Workshop on Research Issues on Data Mining and Knowledge Discovery*, 1997.
13. J. Han, Y. Cai and N. Cercone. Knowledge discovery in databases: an attribute oriented approach. In *Proc. of the 18th Conf. on Very Large Data Bases (VLDB)*, 1992, pp. 547–559.
14. H.V. Jagadish, A. O. Mendelzon and T. Milo. Similarity-based queries. In *Proc. of 14th Symp. on Principles of Database Systems (PODS)*, 1995, pp. 36–45.
15. Y. Karov and S. Edelman. Similarity-based word sense disambiguation. *Computational Linguistics*, 24(1), 1998, pp. 41-59.
16. J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. 9th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1998.
17. A. J. Knobbe and P. W. Adriaans. Analyzing binary associations. In *Proc. of the 2nd Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, 1996, pp. 311-314.
18. R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In *Proc. of VLDB*, 1994, pp. 144–155.
19. D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. *SIGMOD Record*, 26(2), 1997, pp. 13–25.
20. R. Srikant and R. Agrawal. Mining generalized association rules. In *Proc. of the 21st Intl. Conf. on Very Large Data Bases (VLDB)*, 1995, pp. 407–419.
21. G. Strang. Linear algebra and its applications. Harcourt Brace International Edition, 1988.
22. D. A. White and R. Jain. Algorithms and strategies for similarity retrieval. Technical Report VCL-96-101, Visual Computing Laboratory, UC Davis, 1996.
23. T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An efficient data clustering method for very large databases. In *Proc. of SIGMOD*, 1996, pp. 103–114.