Clustering Transactions Using Context-based Distance

Lu, Ke
Graduate School of Economics, Kyushu University

https://doi.org/10.15017/25130

出版情報：経済論究. 143, pp.123-138, 2012-07. 九州大学大学院経済学会
バージョン：
権利関係：

Clustering Transactions Using Context-based Distance

Ke Lu†

Abstract

Transaction clustering is usually the first step towards customer analysis and helps companies to make strategic plans. This paper assumes that a customer transaction contains some products that are called items, and each item belongs to one and only one category based on a product hierarchy. Transaction clustering refers to the data described by categorical attributes and is a challenging task in data mining applications. Unlike numerical attributes, it is difficult to define a distance between values of a categorical attribute, since the values are not ordered. This paper proposes an approach named ***CBDT*** to learn context-based distance for transactions. The main intuition of this work is that the distance between two items in the same category can be determined based on the distribution of items in the other categories, and a transaction can be reformed to be expressed as a tuple. Based on the distance between items, this framework also proposes the distance calculation method for categories and transactions. This approach is validated by embedding this distance learning framework in a hierarchical clustering algorithm.

Keywords: Transaction Clustering, Context-based Distance

Index

† Graduate School of Economics, Kyusyu University, Japan

1 . Introduction

Intense commercial competition induces companies to pay more attention to understanding their customers more deeply to support decision making. Knowledge regarding what customers think, what they want, and how to serve them is quite useful for companies that wish to generate suitable strategies in competitive markets (Hsu et al., 2012). For example, e-commerce companies usually offer distinct home pages and recommend relative products to customers based on predictive models built on customer data. Most financial companies construct their own risk models based on the analysis of customer data to prevent customer credit risk.

Customer data is the corner stone of customer segmentation, and can be briefly separated into two categories. The one is demographic data, which is relatively static in long term. Demographic data may include customers' natural properties, *e.g.*, age and gender, or social properties, *e.g.*, marital status and income. The other one is transaction data, which is relatively dynamic compared with demographic data. Generally, transaction data may include information in purchasing action. Other types of data, *e.g.*, lifestyle data, psychographic data and marketing action data, can be derived from demographic data and transaction data through some statistic methods. Transaction data is merely available for current customers, so that it is necessary to utilize demographic characteristics that are observable in advance for targeting potential customers who are similar to current customers. However, demographic data just plays an auxiliary role in constructing derived data type more often than not. It has been found that transaction data is the most powerful and reliable data for predicting future customer purchase behavior (Yen et al., 2006). This paper focuses on the issues of segmenting customers based on transaction data and the issues related to demographic data are not included.

Data mining techniques have been widely applied in customer relationship management (*CRM*) (Ngai et al., 2009). As an important topic in CRM, customer segmentation, which is based on analysis of customer similarity, has drawn increasing attention, and transaction clustering is usually the first step towards customer segmentation. While clustering is highly expected to help companies understand their customers, it does not seem that existing analysis methods work well enough. Consumers receive significant amount of mails recommending products that they are not interested in (Yang et al., 2003), and online recommendations are still far from acceptable. An important reason is that customers are segmented improperly due to unsuitable similarity measures.

Considerable efforts in finding appropriate distance measures for transaction data have been conducted throughout different applications, because distance measures are fundamentally impor-

tant for clustering data. However, such endeavors calculate the distance between transactions based on the co-occurrence of items. These approaches may face a predicament of differentiation dilemma and overlook the relationship between individual items. Nowadays, companies differentiate their products to tackle the problem of homogenization, so that the total kinds of items in transactions are doubled in the past decades while different items may denote very similar products or highly related products. A motivation example is given as follows.

Example 1.

For three transactions T_1={*Coka-Cola*, *ham*, *tomato*}, T_2={*Pepsi-Cola*, *ham*, *cabbage*} and T_3 ={*Pepsi-Cola*, *ham*, *potato*}, it is said that T_1 is similar to T_2 based on co-occurrence, because both *Coka-Cola* and *ham* are contained in T_1 and T_2, which covers 50% proportion of the total kinds of items of T_1 and T_2. While for T_1 and T_3, there is only one co-occurrence item, which covers only 20% proportion of the total kinds of items in T_1 and T_3, so that T_1 and T_3 cannot be said similar to each other based on co-occurrence, even *Coka-Cola* and *Pepsi-Cola* are both soft drink and very similar to each other.

Generating items seems to overcome this problem, *e.g.*, *Coka-Cola* and *Pepsi-Cola* are generated into soft drink category, and T_1, T_2 and T_3 are similar to each other at the same extent. However, it also causes some new problems, *e.g.*, the detailed information about the similarity would be lost. If the similarity between *Coka-Cola* and *Pepsi-Cola* is also taken into consideration without generation, T_1 and T_2 can be said to be similar to each other to some extent, and the similarity between them could be qualified. □

This example arouses us to take the category relation into considering the similarity between transactions and the distance between two items can be determined by distributions of other items in the transaction database. However, generating items of transactions into higher level in a product hierarchy, *i.e.*, into certain categories, does not seem to help analyzing the similarity between transactions.

This paper proposes an alternative approach ***CBDT*** (Category-Based Distance for Transactions) to evaluate the similarity between transactions based on category information. As a base of this approach, a context-based method is introduced to calculate the distance between individual items. The rest of this paper is organized as follows. Section 2 discusses preliminary problems related to this paper and gives description about the data mentioned in this paper. The issue of calculating distances between transactions is discussed in Section 3. Section 4 discusses the approach to cluster transactions based on distance between transactions and gives a simple example to explain the details of our approach. Section 5 presents the results of a comprehensive set of experiments between our approach and other approach for different datasets. The

conclusion of this paper is presented in Section 6.

2 . Preliminaries

Segmenting customers based on transaction data has been a long overdue issue for a public debate. Motivated by (Papadimitriou et al., 1998), the so called Customer-Oriented Catalog Segmentation problem, which concerns the problem of segmenting customer based on transactions, has been discussed in (Amiri 2006; Ester et al., 2004). On this issue, a customer is covered by a product catalog if n products purchased by this customer belong to that product catalog. The enterprise wants to design k product catalogs of size r that maximize the overall number of customers that are covered by k catalogs. As an important association study field, clustering transactions has drawn increasing attention, and most of these approaches cluster transactions based on the co-occurrence of items (Yang et al., 2003; Yun et al., 2003).

Customer data is the first word to cluster transactions. Transactions are usually stored in form of categorical attributes, and every item is related to a distinct categorical attribute. As a related research field, clustering categorical data has drawn increasing attention. One of the early works in the field of categorical clustering is *K-MODES* (Huang 1998). A cluster is represented as a single instance, or data point, in which each attribute assumes to have the most frequent value in the database. Therefore, in *K-MODES* the similarity of an unlabeled data point and a cluster representative can be simply computed by the overlap distance (Kasif et al. 1998). Another approach to categorical clustering is *ROCK* (Guha et al. 1999). It employs links between pairs of data points in order to measure their similarity. An instance is deemed as the neighborhood of another instance if the Jaccard similarity between them exceeds a threshold given beforehand. It heuristically optimizes a cluster quality function with respect to the number of links between the cluster members. *CACTUS* (Ganti et al. 1999) is a combinatorial search-based algorithm employing summary information of the dataset. Most of the approaches for clustering categorical data are based on co-occurrence of items (or the values of categorical attributes) without using any considering the distance between individual items.

High-dimensional problem is one of the challenges for clustering categorical data. Against this problem, several researchers have proposed to take into consideration the concept of hierarchy to increase the counts of items. A hierarchy is an arrangement of objects in which the objects are represented as being “above,” “below,” or “at the same level” as one another. Attribute Oriented Induction is proposed to reduce the number of items by replacing low level items in concept hierarchy with higher level items to increase the number of occurrences (Han et al., 1992). Multiple-Level Association rule is proposed to count the occurrence of items at a higher level with the summation of item counts at a lower level to increase the counts of items

at higher levels {Han et al, 1999}. Although some patterns or rules can be derived with these approaches and the time consuming is reduced, a detailed level of knowledge is discarded. Segmenting customers by transaction data with concept hierarchy, which gives enlightenment for this paper, have been addressed in (Hsu et al., 2012; Wang et al., 2007). However, former literature only employs concept hierarchy to calculate the distance between items based on the edges between two nodes in a hierarchy.

For the analysis for transaction data in this paper, the transaction data is described in a reformed way given as follows. Let a customer transaction database $\boldsymbol{TDB}=\{T_1, T_2, \cdots, T_l\}$ contain all of transactional records of customers. Let $\boldsymbol{I}=\{i_1, i_2, \cdots, i_r\}$ be the set of product items included in $\boldsymbol{TDB}$, where $i_k(1 \leqslant k \leqslant r)$ is the identifier for the k_{th} item. For two items i_1 and i_2, the distance between them is denoted by $d_{Item}(i_1, i_2)$. A transaction is a subset of $\boldsymbol{I}$. This paper assumes an item belongs to one and only one category, and the categories of $\boldsymbol{TDB}$ are denoted as $\boldsymbol{C}=\{c_1, c_2, \cdots, c_n\}$. Let $c_i(\boldsymbol{I})$ denote the items that belong to category c_i. It is obvious that if $i \neq j$ $c_i(\boldsymbol{I}) \cap c_j(\boldsymbol{I})=\emptyset$, and $\cup_{ci \in C} c_i(\boldsymbol{I})=\boldsymbol{I}$. Let $t_{jk}=T_j \cap c_k(\boldsymbol{I})$. A transaction T_j can be reformed to be denoted by $T^C_j=\{t_{j1}, t_{j2}, \cdots, t_{jn}\}$, where t_{jk} $(1 \leqslant k \leqslant n)$, which is called a cell for transaction T_j, denotes the items belonging to category c_k in $\boldsymbol{C}$ for transaction T_j. By this way, a transaction can be expressed as a tuple with n attributes corresponding to categories. In the rest of this paper, a transaction is denoted as the reformed one and consists of transaction cells corresponding to n attributes. The similarity between transactions can be evaluated by calculating the distance between the tuples. When data are described by tuples, there is a wide range of possible choices to calculate the distance them and distance-based clustering approaches can be easily applied to those data. Euclidean distance, which is a special case of Minkowski distance (also called p-norm distance), may be probably the most popular distance measure.

While transactions are deemed as tuple in this paper, the problem of calculating distance between individual items refers to the distance between categorical data. Compared with data in tuple form, it is complicated to calculate the distance between categorical data, because it is even impossible to rank or compute the distance between two values of the categorical attribute. The details of method to calculate distance between individual items is given in Section 3. In this paper, the concept of distance, both distance between transactions and distance between items, obeys the following mathematical meaning of distance.

Definition 1. Given a set $\boldsymbol{S}$, a real-valued function $d(x, y)$ on the Cartesian product $\boldsymbol{S} \times \boldsymbol{S}$ is a distance if for any $x, y \in \boldsymbol{S}$, it satisfies the following conditions:

(1) $d(x, y) \geqslant 0$ non-negativity
(2) $d(x, y)=d(y, x)$ symmetry
(3) $d(x, y)=0$ if and only if $x=y$ self-identity □

3 . Distance between Transactions

This section introduces the details of the ***CBDT*** approach. The key intuition of this measure is that customers may usually buy the products belonging to different categories together while buy products of the same category seldom. The similarity between transactions is considered on the category level and the transaction level, separately. On the category level, the similarity between individual items should be taken into consideration, while on the transaction level the similarity between transactions should be based on the distance between categories. This approach is based on three stages.

(1) Calculating the distance between individual items.

At this stage, a context-based distance measure is proposed to calculate the distance between individual items that belong to the same categories.

(2) Calculating the distance between corresponding cells of different transactions.

At this stage, a transaction is reformed to be expressed by a tuple by generating the items into the cells corresponding to the category to which they belong. By this way, calculating the distance between corresponding cells refers to the problem calculating distance between sets, because the corresponding categories of different transactions may contain different items.

(3) Calculating the distance between transactions.

After calculating the distance between corresponding categories, calculating distance between transactions is deemed as the problem of calculating distance between two vectors.

3.1 Distances between Individual Items in the Same Category

The items in transactions can be organized by certain product hierarchy, *e.g.*, *Coka-Cola* and *Pepsi-Cola* can be generated into the subcategory *soft drink*, and then *soft drink* and other subcategories can be generated into a higher level in a hierarchy. Even customers may not pay attention to the relation between items based on a hierarchy, in a transaction most of the items belong to different categories, while few items belong to the same category. For example, a customer may like to have *Cola* and *ham* together for a simple meal, while he/she may prefer either *Coka-Cola* or *Pepsi-Cola*. So that, in a transaction, *ham* may appear with *Coka-Cola* or *Pepsi-Cola* together, while *Coka-Cola* and *Pepsi-Cola* may belong to the same transaction seldom. The above discussion induces us that when we compare the similarity between transactions, we should treat the items belonging to the same category and that belonging to different categories distinctly.

Suppose items in a transaction database ***TDB*** belong to m categories and a transaction T_j is

denoted as $T_j=\{c_{j1}, c_{j2},\cdots, c_{jm}\}$, where c_{jk} $(1\leqslant k\leqslant m)$ denotes the items belonging to k_{th} category in a hierarchy for transaction T_j and $c_{jk1}\cap c_{jk2}=\emptyset$ if $k_1\neq k_2$. Here, if a transaction is deemed as a vectorial data with m categorical attributes, categories in a transaction can be treated as categorical attributes and an item in one category is a value for that category. The cardinality of an attribute c_{jk} is denoted as $|c_{jk}|$.

Let the support of item i_l $sup(i_l)$ in a transaction data ***TDB*** be the proportion of transactions that contain item i_l to all of the transactions in ***TDB***. The probability of item i_1 contained in ***TDB*** given that item i_2 contained in ***TDB*** as well, is given as $p(i_1|i_2)=sup(i_1\ i_2)/sup(i_2)$. The distance between two items $i_1, i_2\in c_l$, where c_l is a category, can be determined by the distributions of items belonging to other categories. The corner stone of our approach consists in computing the distance between each pair of items belonging to the same category and given formally as follows.

Definition 2. Let $\boldsymbol{C}$ be the set containing all categories in transaction database ***TDB***. For two items $i_1, i_2\in c_l$, where c_l is a category in $\boldsymbol{C}$, the distance between them is given as follows.

$$d_{Item}(i_1, i_2)=\frac{\sqrt{\sum_{cj\in C/cl}\sum_{ik\in cj}(p(i_1\mid i_k)-p(i_2\mid i_k))^2}}{\sum_{cj\in C/cl}|cj|}$$ □

For two items $i_1, i_2\in c_l$, where c_l is a category, the conditional probabilities of each item $i_k\in c_j(c_j\neq c_l)$ for both items i_1 and i_2 and the Euclidean distance are computed. The Euclidean distance is normalized by the total number of considered items.

This distance measure is an application of the Euclidean distance and obeys the definition of distance given by Definition 1. It is obvious that the value of this distance measure is bounded between 0 and 1: $0\leqslant d_{Item}(i_1, i_2)\leqslant 1$. This distance definition given above strongly depends on the context associated to each item. Here, the context for an item i is assumed to be the distributions of items belonging to transactions that do not contain item i. In practical application, users are free to determine an application-specific context for each item, exploiting his/her knowledge on the domain. Several heuristic automatic data-driven methods to determine context for each item are studied in (Ienco et al., 2012). The algorithm to calculate the distance between individual items is given by Algorithm 1 as follows.

Algorithm 1 ItemDist

Input:

(1)Transaction Database $\boldsymbol{TDB}=\{T^C_1, T^C_2, \cdots, T^C_l\}$.

(2)Category Set $\boldsymbol{C}$.

Output:

Distance Matrix between Items D_{ITEM}.

Method:

(1)**for** each category $\boldsymbol{C}_r$ in $\boldsymbol{C}$ **do**

(2) $\boldsymbol{C}_{temp}=\boldsymbol{C}$;

(3) **for** the items $i_m, i_n \in \boldsymbol{c}_r \mid i_m \neq i_n$ **do**

(4) $\boldsymbol{C}_{temp}=\boldsymbol{C}/\boldsymbol{c}_r$;

(5) $$D_{Item}[i_m][i_n]=\frac{\sqrt{\sum_{cj\in C_{temp}}\sum_{ik\in cj}(p(i_m \mid i_k)-p(i_n \mid i_k))^2}}{\sum_{cj\in C_{temp}}|cj|};$$

(6) **end**

(7)**end**

(8)**for** the items $i_m \in c_r, i_n \in \boldsymbol{c}_l \mid \boldsymbol{c}_r \neq c_l$ **do**

(9) $D_{Item}[i_m][i_n]=null$;

(10)**end**

The distance between items that belong to different categories are defined as *null,* because those distances are not involved in calculating the distance between categories. As an intuitive explanation, the similarity between *Coka-Cola* and *Pepsi-Cola* can be evaluated because they taste similarly to some extent. However, it is strange to evaluate the similarity between *Coka-Cola* and *ham*.

3.2 Distances between Corresponding Cells of Different Transactions

As discussed in subsection 3.1, comparing the distance between two items that belong to different categories has no significant sense. Similarly, it is no sense to compare the similarity between cells corresponding to different categories for transactions. For example, it is strange to calculate the distance between cells that consist of products in category *soft drink* and *vegetable*, respectively. In this subsection, only the distance between cells corresponding to the same category is discussed, and this problem refers to set distance. The discussion starts by introducing two corresponding distance definitions as follows. The one refers to the distance between an item and a set of items. The other one is for the average distance between two sets

of items.

Definition 3. Let i_1 be an item and S a set of items. The distance between them is defined as follows.

$$d_{SI}(i_1, S) = \min_{ik \in S} d_{Item}(i_1, i_k)$$

□

Definition 4. Let S_1 and S_2 be two set of items. If $|S1| \times |S2| \neq 0$, the Sum of Minimum Distance between sets S_1 and S_2 is given as follows.

$$d_{SS}(S_1, S_2) = \frac{\sum_{ij \in S_1} d_{SI}(i_j, S_2) + \sum_{ij \in S_2} d_{SI}(i_k, S_1)}{|S_1 + S_2|}$$

□

This distance measure guarantees that every item in one set is compared with the most similar one in the other set, and the more common items there are between two sets, the shorter the distance between two transactions is. The method to calculate the distance between corresponding cells of different transactions is given by Algorithm 2 as follows.

Algorithm 2 CategoryDist

Input:

(1)Category Set $\boldsymbol{C}$.

(2)Transactions $T^C{}_1$ and $T^C{}_2$: transaction $T^C{}_i (i=1, 2)$ is in form of $T^C{}_i = \{t_{i1}, t_{i2}, \cdots, t_{i|C|}\}$ where $t_{ij} (j=1, 2, \cdots, |\boldsymbol{C}|)$ are the sets of items belong to category c_k in $\boldsymbol{C}$.

Output:

Cell Distance for $T^C{}_1$ and $T^C{}_2$ $d_{Cell}(T^C{}_1, T^C{}_2)$.

Method:

(1)**for** $\boldsymbol{i}=1$ **to** $|C|$
(2) **switch** some conditions of $|t_{1i}|$ and $|t_{2i}|$
(3) **case** $|t_{1i}|+|t_{2i}|=0$ or $t_{1i}=t_{2i}$|their elements are the same
(4) $d_{Cell}(t_{1i}, t_{2i})\ [i]=0$;
(5) **case** $|t_{1i}|+|t_{2i}| \neq 0$ and $|t_{1i}| \times |t_{2i}|=0$
(6) $d_{Cell}(t_{1i}, t_{2i})\ [i]\ =1$;
(7) **case** $|t_{1i}| \times |t_{2i}| \neq 0$
(8) $d_{Cell}(t_{1i}, t_{2i})\ [i]=\ d_{SS}(t_{1i}, t_{2i})$;
(9) **end**
(10)**next** $\boldsymbol{i}$

3.3 Distances between Transactions

After the distance between each corresponding cells of two transactions is given calculated by Algorithm 2, the distance between transactions, which is an application of Euclidean distance, is given as follows.

Definition 4. Let two transactions T^C_1 and T^C_2 consists of n cells corresponding to the product categories. The distance between transactions T^C_1 and T^C_2 is given as

$$d_{Transaction}(T^C_1,\ T^C_2)=\frac{\sqrt{\sum_{i=1}^{n}(d_{Cell}(T^C_1,\ T^C_2)[i])^2}}{n}$$

where $d_{Cell}(T^C_1,\ T^C_2)$ is the category distance vector for T^C_1 and T^C_2. □

The transaction distance matrix is given by Algorithm 3 given as follows.

Algorithm 3 TransactionDist

Input:

Cell Distance Vectors for Transactions in $\boldsymbol{TDB}=\{T^C_1,\ T^C_2,\ \cdots,\ T^C_r\}$.

Output:

Distance Matrix between Transactions $D_{Transaction}$.

Method:

(1)**for** $i=1$ **to** r
(2)　**for** $j=i+1$ **to** r
(3)　　$D_{Transaction}[i][j]=d_{Transaction}(T^C_i,\ T^C_j)$;
(4)　**next** j
(5)**next** i

4. Transaction Clustering

This section discusses the approach to cluster transactions based on distance between transactions. At the end of this section, an illustrative example is given to show the whole approach from calculating the distance between individual items to clustering transactions.

4.1 Clustering Transactions based on Distance between Transactions

This subsection coupled ***DCBT*** with a standard agglomerative (bottom up) hierarchical clustering method. Agglomerative hierarchical clustering algorithms can be applied to create a hierarchy of clusters grouping similar transactions. Clustering starts with a set of singleton clusters, each containing a single transaction in $\boldsymbol{TDB}=\{T^C_1,\ T^C_2,\ \cdots,\ T^C_l\}$. The two most similar transac-

tions are merged to form a new cluster that covers both former clusters (transactions). This process is repeated for each of the remaining n-1 clusters, where n is the number of clusters before each merging stage. There are several distance measures can be used to merge two clusters by evaluating the distance between individual transactions. In this paper, complete linkage distance is employed to determine the overall similarity of clusters based on the transaction similarity matrix. Merging of transactions clusters continues until a single, all-inclusive cluster remains. At termination, a uniform, binary hierarchy of transaction clusters is produced.

4.2 An Illustrative Example

Consider the following example. As shown in Table 1, there are eight transactional records and 4 categories in a ***TDB***. There are two items (products) in every category. The first step of our approach is calculating the distance between items. After the Algorithm 1 is conducted, the distance between items in different categories is given in Table 2. Based on the distance matrix shown in table 2, the distance between transactions is given in Table 3 byconducting Algorithm 2.

Based on the distance matrix between transactions, the distance between transactions T_1 and T_2 is taken as an example to show the details of computing distance between transactions. Firstly, transactions T_1 and T_2 are transformed to the $T^c{}_1=\{c_{11}, c_{12}, c_{13}, c_{14}\}$ and $T^c{}_2=\{c_{21}, c_{22}, c_{23}, c_{24}\}$ respectively. For cells c_{11} and c_{21}, either of them is empty and they are not equal to each other. The distance between them is calculated by Sum of Minimum Distance as (0.1219+0.1219)/2. For cells c_{12} and c_{22}, the cell distance between them is 0 because the items in both c_{12} and c_{22} are the same. The situation of c_{13} and c_{23} is similar. For cells c_{14} and c_{24}, the distance between them is 1 because there is an item in cell c_{14} of transaction T_1, while there is not any item in cell c_{24} of transaction T_2. As a result, the distance between transactions T_1 and T_2 is 0.2519. The distance matrix between transactions is given as in Table 3 and by employing hierarchical clustering algorithm, the final clustering result is shown as in Figure 1.

Table 1: Transactional Records in ***TDB***

	Category 1		Category 2		Category 3		Category 4	
	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
$T^c{}_1$	0	1	1	1	0	0	1	0
$T^c{}_2$	1	0	1	1	0	0	0	0
$T^c{}_3$	1	0	0	1	0	1	1	0
$T^c{}_4$	1	1	1	0	1	0	1	0
$T^c{}_5$	0	0	0	0	0	1	0	1
$T^c{}_6$	0	0	0	1	0	1	1	1
$T^c{}_7$	1	0	1	1	1	0	1	1
$T^c{}_8$	1	1	0	0	1	0	1	1

Table 2: Distance between Items of the Same Category

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
i_1	0	0.1219						
i_2	0.1219	0						
i_3			0	0.1450				
i_4			0.1450	0				
i_5					0	0.1601		
i_6					0.1601	0		
i_7							0	0.1768
i_8							0.1768	0

Table 3: Distance between Transactions

	T^c_1	T^c_2	T^c_3	T^c_4	T^c_5	T^c_6	T^c_7	T^c_8
T^c_1	0	0.2519	0.2521	0.2505	0.4353	0.3541	0.2523	0.3540
T^c_2	0.2519	0	0.3538	0.3539	0.5	0.4332	0.3536	0.4331
T^c_3	0.2521	0.3538	0	0.0549	0.3563	0.2539	0.0443	0.0438
T^c_4	0.2505	0.3539	0.0549	0	0.3585	0.2562	0.0216	0.2504
T^c_5	0.4353	0.5	0.3563	0.3585	0	0.2536	0.3561	0.2536
T^c_6	0.3541	0.4332	0.2539	0.2562	0.2536	0	0.2535	0.3558
T^c_7	0.2523	0.3536	0.0443	0.216	0.3561	0.2535	0	0.2534
T^c_8	0.3540	0.4331	0.0438	0.2504	0.2536	0.3558	0.2534	0

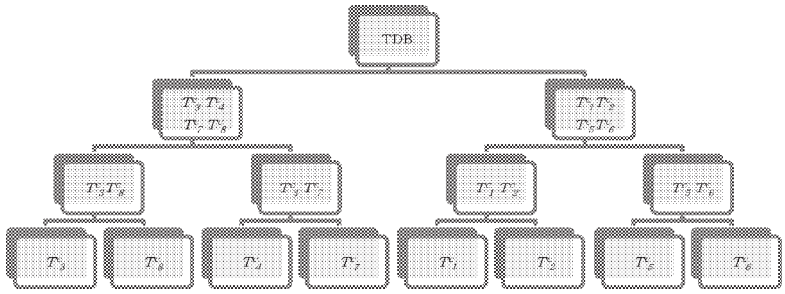

Figure 1: The Result of Clustering Transactions

5 . Comparative Experiments and Results

In this section, the approach proposed in this paper is compared with two former methods in literature on two datasets. Comparison experiment starts by introducing two measures to evaluating the results of clustering.

5.1 Clustering Evaluation Measures

Measuring the quality of clustering is usually a hard and subjective task. In this paper, two

objective criteria are employed to evaluate the clustering results: Purity and Normalized Mutual Information (***NMI*** for short). These methods make use of the correspondence between the original class information of each transaction and the cluster to which the same transactions have been assigned. Let $\boldsymbol{CL}=\{cl_1, cl_2, \cdots, cl_j\}$ denote the partition built by the clustering algorithm on transactions and $\boldsymbol{P}=\{p_1, p_2, \cdots, p_j\}$ the partition inferred by the original classification. j and i *are* the total number of clusters $|\boldsymbol{CL}|$ and the number of classes $|\boldsymbol{P}|$, respectively. l is the total number of transactions in ***TDB***.

The first clustering quality measure is *purity* in terms of the class to which the clustered transaction belongs. In order to compute purity, each cluster is compared with the majority class of the transactions in the cluster. Then, the accuracy of this assignment is measured by counting the number of correctly assigned objects divided by the total number of objects n. The larger the *purity* is, the better the clustering is. The *purity* measure is given as follows

$$purity(C, P)=\frac{1}{n}\sum_{cl_j\in CL}\max_{p_i\in P}|cl_j\cap p_i|$$

It should be noticed that *purity* measure is sensible to the presence of imbalanced classes.

The second measure provides the information that is independent from the number of clusters (Strehl et al. 2002). This measure takes its maximum value when the clustering partition matches completely the original partition. ***NMI*** can be deemed as an indicator of the purity of the clustering result. ***NMI*** is computed as the average mutual information between any pair of clusters and classes. This measure is formally given as follows.

$$NMI=\frac{\sum_{i=1}^{I}\sum_{j=1}^{J}n_{ij}\log\frac{n_{ij}n}{n_i n_j}}{\sqrt{\sum_{i=1}^{I}n_i\log\frac{n_i}{n}\sum_{j=1}^{J}n_i\log\frac{n_j}{n}}}$$

where n_{ij} is the cardinality of the set of objects that occur both in cluster i and in class j; n_i is the number of objects in cluster i; n_j is the number of objects in class j; n is the total number of objects. I and J are the total number of clusters and the number of classes, respectively.

5.2 Datasets for Comparison

Two real world data sets (Mushroom and Voting), which are downloaded from the UCI Machine Learning Repository (Blake et al., 1998), are employed to compare the formal approaches and our approach. Mushroom dataset includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Voting dataset includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the Congressional Quarterly Almanac, 98th Congress 1984. Both datasets are not transaction data in strict term. However, they can be deemed as transaction data by some conceptual transforming because all of the attributes in both datasets are categorical attributes.

If every record is deemed as a transaction and every attribute is regarded as a category, the possible values for each attribute can be considered as an item for the corresponding category (attribute here). The main characteristics of these two datasets are summarized in Table 4.

Table 4: Characteristics of Datasets

Dataset	Instances	Attributes	Values	Classes
Mushroom	8124	22	117	2
Voting	435	16	32	2

5.3 Comparing the Clustering Results of Various Approaches

The comparison experiments are made in two ways. In the first way, our approach is compared with ***ROCK*** (Guha et al., 1999). ***ROCK*** approach is employed to cluster Mushroom and Voting datasets. On the other hand, the distance learning method described in (Ahmad et al., 2007), which is referred as ***DELTA*** in this paper, is employed to calculate the distance between categories (which is actually a categorical attribute in this paper) of the same datasets, and then the distance between transactions are calculated by Algorithm 3. Finally transactions are clustered by standard agglomerative (bottom up) hierarchical clustering method. In this paper, there are two classes in both datasets. Since the agglomerative (bottom up) hierarchical clustering method returns a dendrogram that contains a different number of clusters at each level, only the level that is corresponding to two clusters is taken into consideration. Similarly, for ***ROCK***, the threshold parameter is set to let the clustering result contain two clusters. Both clustering results by these two ways are compared with the clustering result of our approach. The evaluation of clustering results is shown in the Table 5 and Table 6 as follows. As shown in these two tables, the clustering results of ***CBDA*** are better than the other two former approaches for both datasets.

Table 5: Purity Results with respect to Various Approaches

Dataset	***CBDA***	***ROCK***	***DELTA***
Mushroom	89.02%	49.43%	89.02%
Voting	91.95%	83.90%	89.43%

Table 6: ***NMI*** Results with respect to Various Approaches

Dataset	***CBDA***	***ROCK***	***DELTA***
Mushroom	0.5938	0.05681	0.5938
Voting	0.6009	0.3446	0.4994

6 . Conclusions

This paper introduced a context-based distance measure to evaluate the distance between items that belong to the same category. It is believed that the proposed approach is general enough for normal transaction clustering task. Meanwhile it can be applied to any data mining task that involves categorical data and requires computing distance. An example of application is within a nearest neighbors (kNN) classifier to estimate a good distance measure between two instances that contain categorical attributes. Our approach is preferable for Boolean variables as well, because Boolean variable is a special case of categorical attributes. The future work includes investigating the application of this distance measure to different distance-based tasks. The comparative experiments show that the approach proposed in this paper is better than some state of the art approaches. The efficiency evaluation of our approach is included in our future work as well.

References

Ahmad, A. and Dey, L. (2007). A method to compute distance between two categorical values of same ttribute in unsupervised learning for categorical data set. Pattern Recognition Letters 28(1), 1860–1871.

Amiri, A. (2006). Customer-oriented catalog segmentation: effective solution approaches. Decision Support Systems 42(3), 1860–1871.

Blake, C. L. and Merz, C. J. (1998). UCI repository of machine learning databases. http://www.ics.uci.edu/ mlearn/ MLRepository.html.

Ester, M., Ge, R., Jin, W. and Hu, Z. J. (2004). A microeconomic data mining problem: customer-oriented catalog segmentation. In: The 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.557–562.

Ganti, V., Gehrke, J. and Ramakrishnan, R. (1999). Cactus-clustering categorical data using summaries. In Proc. of the 5th International ACM SIGKDD Conference on Knowledge Discovery and Data Mining. ACM, 73–83.

Guha, S., Rastogi, R. and Shim, K. (1999). ROCK: A robust clustering algorithm for categorical attributes. In Proc. of the 15th International Conference on Data Engineering (ICDE). IEEE Computer Society, 512–521.

Han, J. W. and Fu, Y. J. (1999). Mining multiple-level association rules in large databases. IEEE Trans. Knowl. Data Eng. 11(5), 798–804.

Han, J. W., Cai, Y. D. and Nick, C. (1992). Knowledge discovery in databases: an attribute-oriented approach. In Proc. 18th Int. Conf. Very Large Data Bases, 547–559.

Hsu, F. M., Lu, L. P. and Lin, C. M. (2012). Segmenting customers by transaction data with concept hierarchy. Expert Syst. Appl. 39(6), 6221–6228.

Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Min. Knowl. Discov. 2(3), 283–304.

Ienco, D., Pensa, R. G. and Meo, R. (2012). From context to distance: learning dissimilarity for categorical data clustering. The Transactions on Knowl. Discov. from Data. 6(1), 283–304.

Kasif, S., Salzberg, S., Waltz, D. L., Rachlin, J. and Aha, D. W. (1998). A probabilistic framework for memory-based reasoning. Artif. Intell. 104 (1), 287–311.

Ngai, E. W. T., Li, X., and Chau, D. C. K. (2009). Application of data mining techniques in customer relationship management: a literature review and classification. Expert Syst. Appl. 36(2), 2592–2602.

Kleinberg, J. M., Papadimitriou, C. H. and Raghavan, P. (1998). A microeconomic view of data mining. Data Min. Knowl. Discov. 2(4), 311–324.

Wang, M.T., Hsu, P.Y., Lin, K.C. and Chen S.S. (2007). Clustering transactions with an unbalanced hierarchical product structure. In: 12th International Conference on Data Warehousing and Knowledge Discovery, pp.251–261.

Yang, Y. H., and Padmanabhan, B. (2003). Segmenting customer transactions using a pattern-based clustering approach. In: The 3rd IEEE International Conference on Data Mining, pp.411–418.

Yen, S. F., and Lee, Y. S. (2006). An efficient data mining approach for discovering interesting knowledge from customer transactions. Expert Syst. Appl. 30(4), 650–657.

Yun, C. H., Chuang K. T. and Chen, M. S. (2003). Clustering item data sets with association-taxonomy similarity. In: The 3rd IEEE International Conference on Data Mining, pp.697–700.