A Cluster Ensemble Method for Clustering Categorical Data

Zengyou He[*], Xiaofei Xu, Shengchun Deng

Department of Computer Science and Engineering Harbin Institute of Technology,
92 West Dazhi Street, P.O Box 315, P. R. China, 150001

Received XX; received in revised form XX; accepted XX

Abstract Categorical data clustering (CDC) and cluster ensemble (CE) have long been considered as separate research and application areas. The main focus of this paper is to investigate the commonalities between these two problems and the uses of these commonalities for the creation of new clustering algorithms for categorical data based on cross-fertilization between the two disjoint research fields. More precisely, we formally define the CDC problem as an optimization problem from the viewpoint of CE, and apply CE approach for clustering categorical data. Experimental results on real datasets show that CE based clustering method is competitive with existing CDC algorithms with respect to clustering accuracy.

Keywords Clustering, Categorical Data, Cluster Ensemble, Data Mining

1. Introduction

Clustering typically groups data into sets in such a way that the intra-cluster similarity is maximized while the inter-cluster similarity is minimized. The clustering technique has been extensively studied in many fields such as pattern recognition [1], customer segmentation [2], similarity search [3] and trend analysis [4].

Most previous clustering algorithms focus on numerical data whose inherent geometric properties can be exploited naturally to define distance functions between data points. However, much of the data existed in the databases is categorical, where attribute values can't be naturally ordered as numerical values. An example of categorical attribute is *shape* whose values include *circle*, *rectangle*, *ellipse*, etc. Due to the special properties of categorical attributes, the clustering of categorical data seems more complicated than that of numerical data. A few algorithms have been proposed in recent years for clustering categorical data [5-24].

Cluster ensemble (CE) is the method to combine several runs of different clustering algorithms to get a common partition of the original dataset, aiming for consolidation of results from a portfolio of individual clustering results. Although the research on cluster ensemble has not been widely recognized as that combing multiple classifier or regression models, more recently, several research efforts have been done independently [e.g., 25-28].

Until recently, CDC and CE have long been considered as separate research and application areas. The starting point in this paper is the observation of some key underlying similarities between these two different areas. This observation makes possible the study of CDC problem from a CE perspective. This different perspective may enable a better understanding of the CDC

[*] Corresponding author. Tel: +86-451-6414906 (ext. 8512). *Email address*: zengyouhe@yahoo.com (Z. He).

algorithms and help in devising improved or hybrid versions by combining elements from areas that would otherwise be considered incompatible. That is, our first contribution is the exploration of underlying properties, similarities and differences between CDC and CE, which creates the basis for the proposal of CE based clustering algorithms for categorical data. More precisely, although CE is a general framework with many applications and CDC is a special case in clustering research, from a restricted viewpoint, these two problems are equivalent in essence.

Our second contribution is the direct adaptation and use of CE methodology for clustering categorical data. We formally define the CDC problem as an optimization problem from the viewpoint of CE, and apply CE approach for clustering categorical data. Our experimental results show the new categorical data clustering methods to achieve better clustering accuracy than previous algorithms, which confirms our intuition that CE approaches and CDC methods can be used interchangeably. Furthermore, the idea of linking CE and CDC will enable a problem at hand to be solved through either way. Thus, improvements can be achieved in both domains.

The remainder of this paper is organized as follows. Section 2 presents a critical review on related work. Section 3 creates an interesting view on the underlying properties, similarities and differences between CDC and CE. In Section 4, we define the CDC problem as an optimization problem and describe the CE based algorithms for clustering categorical data. Experimental results are given in Section 5 and Section 6 concludes the paper.

2. Related Work

2.1 Clustering Categorical Data

A few algorithms have been proposed in recent years for clustering categorical data [5-24]. In [5], the problem of clustering customer transactions in a market database is addressed. STIRR, an iterative algorithm based on non-linear dynamical systems is presented in [6]. The approach used in [6] can be mapped to a certain type of non-linear systems. If the dynamical system converges, the categorical databases can be clustered. Another recent research [7] shows that the known dynamical systems cannot guarantee convergence, and proposes a revised dynamical system in which convergence can be guaranteed.

K-modes, an algorithm extending the *k*-means paradigm to categorical domain is introduced in [8,9]. New dissimilarity measures to deal with categorical data is conducted to replace means with modes, and a frequency based method is used to update modes in the clustering process to minimize the clustering cost function. Based on *k*-modes algorithm, [10] proposes an adapted mixture model for categorical data, which gives a probabilistic interpretation of the criterion optimized by the *k*-modes algorithm. A fuzzy *k*-modes algorithm is presented in [11] and tabu search technique is applied in [12] to improve fuzzy *k*-modes algorithm. An iterative initial-points refinement algorithm for categorical data is presented in [13]. The work in [23] can be considered as the extensions of *k*-modes algorithm to transaction domain.

In [14], the authors introduce a novel formalization of a cluster for categorical data by generalizing a definition of cluster for numerical data. A fast summarization based algorithm, CACTUS, is presented. CACTUS consists of three phases: *summarization*, *clustering*, and *validation*.

ROCK, an adaptation of an agglomerative hierarchical clustering algorithm, is introduced in

[15]. This algorithm starts by assigning each tuple to a separated cluster, and then clusters are merged repeatedly according to the closeness between clusters. The closeness between clusters is defined as the sum of the number of “links” between all pairs of tuples, where the number of “links” is computed as the number of common neighbors between two tuples.

In [16], the authors propose the notion of *large item*. An item is *large* in a cluster of transactions if it is contained in a user specified fraction of transactions in that cluster. An allocation and refinement strategy, which has been adopted in partitioning algorithms such as *k*-means, is used to cluster transactions by minimizing the criteria function defined with the notion of large item. Following the large item method in [16], a new measurement, called the small-large ratio is proposed and utilized to perform the clustering [17]. In [18], the authors consider the item taxonomy in performing cluster analysis. While the work [19] proposes an algorithm based on “caucus”, which is fine-partitioned demographic groups that is based the purchase features of customers.

Squeezer, a one-pass algorithm is proposed in [20]. *Squeezer* repeatedly read tuples from dataset one by one. When the first tuple arrives, it forms a cluster alone. The consequent tuples are either put into an existing cluster or rejected by all existing clusters to form a new cluster according to the given similarity function.

COOLCAT, an entropy-based algorithm for categorical clustering, is proposed in [21]. Starting from a heuristic method of increasing the height-to-width ratio of the cluster histogram, the authors in [22] develop the CLOPE algorithm. [24] introduce a distance measure between partitions based on the notion of generalized conditional entropy and a genetic algorithm approach is utilized for discovering the median partition.

2.2 Cluster Ensemble

In [25], the authors formally defined the CE problem as an optimization problem and propose combiners for solving it based on a hyper-graph model.

A multi-clustering fusion method is presented in [27]. In that method, the results of several independent runs of the same clustering algorithm are appropriately combined to obtain a partition of the data that is not affected by initialization and overcomes the instabilities of clustering methods. After that, the fusion procedure starts with the clusters produced by the combining part and finds the optimal number of clusters according to some predefined criteria.

The authors in [28] proposed a sequential combination method to improve the clustering performance. First, their algorithm uses the global criteria based clustering to produce an initial result, then use the local criteria based information to improve the initial result with a probabilistic relaxation algorithm or linear additive model.

Other cluster ensemble methods are proposed in [29,30,31].

3. A Unified View on CDC and CE

The researches on CDC and CE have been conducted in parallel. Our goal in this section is to argue that a unified view can be built for the CDC problem and CE problem, hence, CDC problem can be solved with existing CE algorithms.

3.1 Introductory Concepts and Notations

Clustering aims at discovering groups and identifying interesting patterns in a dataset. We call a particular clustering algorithm with a specific view of the data a *clusterer*. Each clusterer outputs a *clustering* or *labeling*, comprising the group labels for some or all objects.

Let $X = \{x_1, x_2 \ldots x_n\}$ denote a set of objects/samples/points. A partitioning of these n objects into k clusters can be represented as a set of k sets of objects $C_l=\{l=1,\ldots, k\}$or as a label vector $\lambda \in N^n$. A clusterer$\Phi$is a function that delivers a label vector given a set of objects. Fig.1 (adapted from [25]) shows the basic setup of the cluster ensemble: A set of r labelings $\lambda^{(1,2,\ldots,r)}$ is combined into a single labelingλ (the *consensus labeling*) using a consensus function Γ. A superscript in brackets denotes an index and not an exponent.

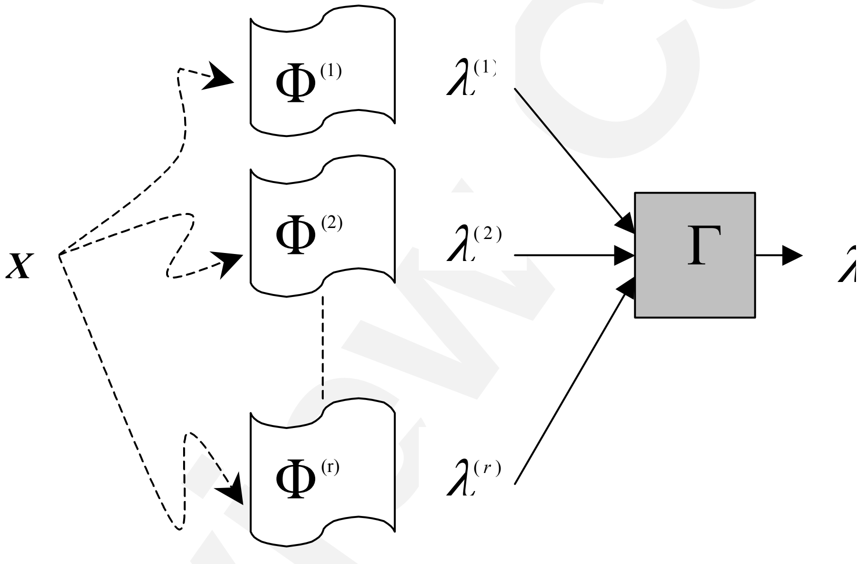

Fig. 1. The cluster ensemble. A consensus function Γ combines clusterings $\lambda^{(q)}$ from a variety of sources.

3.2 A Unified View in the CE Framework

In this section, we firstly discuss the similarities between CDC problem and CE problem from the perspectives of input, output and objective to achieve. Then we present a unified view for the two problems in the CE framework (see Fig. 1).

3.2.1 Similarities

(1) Input: From the viewpoint of clustering, data objects with different cluster labels are considered to be in different clusters, if two objects are in the same cluster then they are considered to be fully similar, otherwise they are fully dissimilar. Thus, it is obvious that cluster labels are impossible to be given a natural ordering in a way similar to real numbers, i.e., the output of clustering algorithm can be viewed as *categorical* (or *nominal*).

Therefore, the input for CE problem is a categorical dataset. That is, in both CE and CDC problems, the datasets to be handled are categorical.

(2) Output: CE tries to combine several runs of different clustering algorithms to get a common partition of the original dataset, aiming for consolidation of results from a portfolio of individual clustering results. Hence, the output for the CE problem is just the same as that of CDC problem.

(3) Objective to achieve: Both CE and CDC aim at grouping the input categorical data into sets in such a way that the intra-cluster similarity is maximized while the inter-cluster similarity is minimized.

Based on the above observations, we can get the conclusion that the CE problem and CDC problem are equivalent. Therefore, algorithms developed in both domains can be used interchangeably, which would enable a problem at hand to be solved through either way. Complementary to our method, we recently learned about two approaches [31, 35] that solves CE problem with a CDC algorithm, which provides evidence on the equivalence of the two problems in a reverse perspective.

3.2.2 A Unified View in the CE Framework

For a categorical dataset, if we consider attribute values as cluster labels, each attribute with its attribute values give a "*best clustering*" on the dataset without considering other attributes. So the CDC problem can be considered as the CE problem, in which the attribute values of each attribute are the outputs of different clustering algorithms.

More precisely, let the dataset $X = \{x_1, x_2 \dots x_n\}$ be a set of objects described by r categorical attributes, $A_1, \dots, A_r$ with domains $D_1, \dots, D_r$ respectively. The value set V_i is a set of values of A_i that are present in X. Recalling the CE framework described in Fig.1, if we define each clusterer $\Phi^{(i)}$ as a function that mapping values in V_i to distinct natural numbers, we can get the optimal partitioning $\lambda^{(i)}$ determined by each attribute A_i as: $\lambda^{(i)} = \{\Phi^{(i)}(x_j.A_i) \mid x_j.A_i \in V_i, x_j \in X\}$. So, we can combine the set of r labelings $\lambda^{(1,2,\dots,r)}$ into a single labeling λ using a consensus function Γ to get the solution for the CDC problem.

For example, Table 1 shows a categorical table with 10 records, each described by 2 categorical attributes. Only considering "*Attribute 1*", we can get the optimal partitioning {(1,2,5,7,10), (3,4,6,8,9)} with 2 clusters. Similarly, "*Attribute 2*" gives an optimal partitioning as {(1,4,9), (2,3,10), (5,6,7,8)} with 3 clusters. Then, we can use the cluster ensemble approach to combine the 2 partitionings and hence get the final clustering output for the categorical dataset.

Furthermore, considering the CDC and CE problems in a unified view may enable a better understanding of their natures, and improvements can be achieved in both domains.

Table 1 Sample Categorical Data Set

Record Number	***Attribute 1***	***Attribute 2***
1	M	A
2	M	B
3	F	B
4	F	A
5	M	C
6	F	C
7	M	C
8	F	C
9	F	A
10	M	B

3.3 Differences

It's the time to mention the differences between the CDC and CE problem. Besides their difference in concepts, as we have discussed in Section 3.2, they are the same problem in nature. However, it should be noted that they *do* have slight difference in their *input*.

In general, no (or only a few) duplicates exist the input categorical dataset for the CDC algorithms. While the input categorical dataset for the CDC problem commonly contains a large amounts of duplicated objects because the *clusterers* often produce clusterings that are similar to each other.

Moreover, most proposed algorithms for CDC problem deserve good scalabilities, because data mining person mainly conducts research in this field. In contrast, most CE algorithms focus on producing good clustering outputs and don't care too much about the execution time.

4. Cluster Ensemble Based Approach

In this section, we borrow the idea of cluster ensemble [25, 26] to formalize the CDC problem as an optimization problem in terms of shared mutual information and describe those CE based algorithms for clustering categorical data.

4.1 Object Function for CDC

Consider the dataset $X = \{x_1, x_2\ldots x_\mathrm{n}\}$ be a set of objects described by r categorical attributes, $A_1, \ldots, A_r$ with domains $D_1,\ldots, D_r$ respectively. The value set V_i is a set of values of A_i that are present in X. As pointed out in Section 3.2, if we define each clusterer $\Phi^{(i)}$ as a function that mapping values in V_i to distinct natural numbers, we can get the optimal partitioning $\lambda^{(i)}$ determined by each attribute A_i. Hence the final clustering output can be regarded as the cluster ensemble result by combining the clusters given by $\lambda^{(i)}$.

Intuitively, a good combined clustering should share as much information as possible with the given *r* labelings. Strehl and Ghosh [25,26] use the mutual information in information theory to measure the shared information, which can be directly applied in this literature.

More concisely, as shown in Strehl's papers [25,26], given *r* groupings with the *q*-th grouping $\lambda^{(q)}$ having $k^{(q)}$ clusters, a consensus function Γ is defined as a function $N^{n\times r} \to N^{n}$ mapping a set of clusterings to an integrated clustering:

$$\Gamma : \{\lambda^{q} \mid q \in \{1,2,...,r\}\} \to \lambda \qquad (1)$$

The set of groupings is denoted as $\Lambda = \{\lambda^{q} \mid q \in \{1,2,...,r\}\}$. The optimal combined clustering should share the most information with the original clusterings. In information theory, mutual information is a symmetric measure to quantify the statistical information shared between two distributions. Let *A* and *B* be the random variables described by the cluster labeling $\lambda^{(a)}$ and $\lambda^{(b)}$, with $k^{(a)}$ and $k^{(b)}$ gruops respectively. Let *I* (*A*, *B*) denote the mutual information between *A* and *B*, and *H* (*A*) denote the entropy of *A*. As Strehl has shown in [26], $I(A,B) \le \dfrac{H(A)+H(B)}{2}$ holds. Hence, the [0,1]-normalized mutual information (*NMI*)[1] [26] used is:

$$NMI(A,B) = \frac{2I(A,B)}{H(A)+H(B)} \qquad (2)$$

Obviously, *NMI* (*A*, *A*) = 1. Equation (2) has to be estimated by the sampled quantities provided by the clusterings [26]. As shown in [26], if we let $n^{(h)}$ be the number of objects in cluster C_h according to $\lambda^{(a)}$, and let n_g be the number of objects in cluster C_g according to $\lambda^{(b)}$. Let $n_g^{(h)}$ be denote the number of objects in cluster C_h according to $\lambda^{(a)}$ as well as in cluster C_g according to $\lambda^{(b)}$. The [0,1]-normalized mutual information criteria $\phi^{(NMI)}$ is computed as follows [25,26]:

$$\phi^{(NMI)}(\lambda^{(a)},\lambda^{(b)}) = \frac{2}{n}\sum_{h=1}^{k^{(a)}}\sum_{g=1}^{k^{(b)}} n_g^{(h)} \log_{k^{(a)}*k^{(b)}}\left(\frac{n_g^{(h)} n}{n^{(h)} n_g}\right) \qquad (3)$$

Therefore, the Average Normalized Mutual Information (*ANMI*) between a set of *r* labelings, Λ, and a labeling $\tilde{\lambda}$ is defined as follows [26]:

[1] In the more recent work of Strehl and Ghosh [36], the authors use a different definition of (A)NMI. The source code that is available on the web and that we use also uses that new definition.

$$\phi^{(ANMI)}(\Lambda,\tilde{\lambda}) = \frac{1}{r}\sum_{q=1}^{r}\phi^{(NMI)}(\tilde{\lambda},\lambda^{(q)}) \tag{4}$$

According to [25,26], the optimal combined clustering $\lambda^{(k-opt)}$ should be defined as the one that has the maximal average mutual information with all individual partitioning $\lambda^{(q)}$ given that the number of consensus clusters desired is *k*. Thus the objective function for categorical data clustering is Average Normalized Mutual Information (*ANMI*). Then, it is defined as [26]:

$$\lambda^{(k-opt)} = \arg\max_{\tilde{\lambda}}\sum_{q=1}^{r}\phi^{(NMI)}(\tilde{\lambda},\lambda^{(q)}) \tag{5}$$

where $\tilde{\lambda}$ goes through all possible *k*-partitions.

As noted in [25, 26], more balanced clusters are desired in the object function presented in Equation (5), which is also observed in our experiments. This is a good property since many real life data mining applications demand comparably sized segments of the data, irrespective of whether the natural clusters in the data have balanced sizes or not.

Since we have pointed out that the CDC problem can be considered as a CE problem. So, using Equation (5) as an object function to be maximized, we formally define the CDC problem as an optimization problem. Compared with other optimization models in this field, such as [9,16, 21, 24], our formalization is more intuitive and suitable for the categorical data from an optimization aspect.

4.2 Cluster Ensemble Based Algorithms

So far, there are several algorithms for cluster ensemble [e.g., 25-28]. The approach in [27] is designed for combining runs of clustering algorithms with the same number of clusters. Thus, it is not suitable in our literature, for the number of clusters determined by different categorical attribute can be different. The sequential combination method proposed in [28] has the same problem as the approach in [27]. In addition, their algorithm has the limitation to combine only the outputs of *two specific* clustering algorithms.

Strehl and Ghosh [25, 26] propose three hypergraph-model based algorithms, namely CSPA, HGPA and MCLA for cluster ensemble, which are adopted for clustering categorical data in this paper. In the following, we will give brief introductions on the three algorithms.

(2.1) CSPA

If two objects are in the same cluster then they are considered to be fully similar, and if not they are dissimilar. This is the simplest heuristic and is used in the Cluster-based Similarity Partitioning Algorithm (**CSPA**) [25]. With this viewpoint, one can simply reverse engineer a single clustering into a binary similarity matrix. Similarity between two objects is 1 if they are in the same cluster and 0 otherwise. For each clustering, a binary similarity $n \times n$ matrix is created. The entry-wise average of *r* such matrices representing the *r* sets of groupings yield an overall similarity matrix. Then, the METIS [32] algorithm is used to partition the similarity graph (vertex= object, edge weight = similarity) to get the final clusters.

(2.2) HGPA

Each cluster is represented as a hyperedge with the same weights, the data objects are considered as vertices with the same weights. Then, a hypergraph partitioning algorithm, HMETIS [33], is used to partition the hypergraph such that the sum of weights hyperedge cut is minimized. The produced unconnected components are taken as the final outputs.

(2.3) MCLA

As done in HGPA, each cluster is represented as a hyperedge. The idea in MCLA is to group and collapse related hyperedges and assign each object to the collapsed hyperedge in which it participates most strongly. The hyper-edges that are considered related for the purpose of collapsing are determined by a graph based clustering of hyperedges. Each cluster of hyperedges is referred as a meta-cluster [26]. Collapsing reduce the number of hyper-edges to *k*.

Since the objective function (Equation (5)) has an added advantage that it allows one to add a stage that selects the algorithm without any supervision information, by simply selecting the one with the highest ANMI [26]. So, for the experiments in this paper, to test the effectiveness of CE method for clustering categorical data, we first run all the three algorithms, CSPA, HGPA and MCLA, and selecting the one with the greatest ANMI as the final result. We denote this integrated CE approach as *ccdByEnsemble* (**C**lustering **C**ategorical **D**ata **By** Cluster **Ensemble**).

5. Experimental Results

A comprehensive performance study has been conducted to evaluate our method. In this section, we describe those experiments and the results. We ran our algorithm on real-life datasets obtained from the UCI Machine Learning Repository [34] to test its clustering performance against other algorithms.

5.1 Real Life Datasets and Evaluation Method

We experimented with four real-life datasets: the Congressional Votes dataset, the Wisconsin Breast Cancer dataset, the Mushroom dataset and the Zoo dataset, which were obtained from the UCI Machine Learning Repository [34]. Now we will give a brief introduction about these datasets.

- ✓ **Congressional Votes:** It is the United States Congressional Voting Records in 1984. Each record represents one Congressman's votes on 16 issues. All attributes are Boolean with Yes (denoted as *y*) and No (denoted as *n*) values. A classification label of Republican or Democrat is provided with each record. The dataset contains 435 records with 168 Republicans and 267 Democrats.
- ✓ **Wisconsin Breast Cancer Data[2]:** It has 699 instances with 9 attributes. Each record is labeled as *benign* (458 or 65.5%) or *malignant* (241 or 34.5%). In our literature, all attributes

[2] We use a dataset that is slightly different from its original format in UCI Machine Learning Repository, which has 683 instances with 444 benign records and 239 *malignant* records. It is public available at: http://research.cmis.csiro.au/rohanb/outliers/breast-cancer/brcancerall.dat.

are considered categorical with values 1,2, …, 10.

- ✓ **The Mushroom Dataset:** It has 22 attributes and 8124 records. Each record represents physical characteristics of a single mushroom. A classification label of poisonous or edible is provided with each record. The numbers of edible and poisonous mushrooms in the dataset are 4208 and 3916, respectively.
- ✓ **The Zoo data** consists of 101 instances of animals with 17 features and 7 output classes. The name of the animal constitutes the first attribute. There are 15 boolean features corresponding to the presence of hair, feathers, eggs, milk, backbone, fins, tail; and whether airborne, aquatic, predator, toothed, breathes, venomous, domestic, catsize. The character attribute corresponds to the number of legs lying in the set {0, 2, 4, 5, 6, 8}.

Validating clustering results is a non-trivial task. In the presence of true labels, as in the case of the data sets we used, the clustering accuracy for measuring the clustering results was computed as follows. Given the final number of clusters, *k*, clustering accuracy *r* was defined as: $r = \frac{\sum_{i=1}^{k} a_i}{n}$, where *n* is the number of records in the dataset, a_i is the number of instances occurring in both cluster *i* and its corresponding class, which had the maximal value. In other words, a_i is the number of records with the class label that dominates cluster *i*. Consequently, the clustering error is defined as $e = 1-r$.

5.2 Experiment Design

We studied the clustering found by three algorithms, our algorithm denoted as *ccdByEnsemble*, the *Squeezer* algorithm introduced in [20] and the GAClust algorithm proposed in [24]. Choosing the *Squeezer* algorithm and GAClust algorithm for comparison is based on the following considerations.

It has been demonstrated that the *Squeezer* algorithm [20] can produce better clustering output than other algorithms in categorical dataset with respect to clustering accuracy. Thus, this algorithm is selected for the competition. In [24], the CDC problem is also formalized as an optimization problem based on information theory, which is similar to our method while they use a very different object function. Hence, comparing our method with the GAClust algorithm [24] will provide us an insight on the advantage of our mutual information based formalization for the CDC problem.

Until now, there is no well-recognized standard methodology for CDC experiments. However, we observed that most clustering algorithms require the number of clusters as an input parameter, so in our experiments, we cluster each dataset into different number of clusters, varying from 2 to 9. For each fixed number of clusters, the clustering errors of different algorithms were compared.

In all the experiments, except for the number of clusters, all the parameters required by the *ccdByEnsemble* algorithm are set to be default[3]. The *Squeezer* algorithm requires *only* a similarity threshold as input parameter, so we set this parameter to a proper value to get the desired number

[3] Since our implementation for the *ccdByEnsemble* algorithm is adapted from ClusterEnsemble algorithms developed by Strehl [25,26, 36]. So, the readers may refer to Strehl's codes for implementation details. The source codes of 'ClusterEnsemble' are available at http://www.strehl.com/.

of clusters (For the *Squeezer* algorithm, if the output number of clusters is same, the clustering accuracy is almost identical. Hence, we can use *any* similarity threshold value that can make the algorithm get the desired number of clusters). For the GAClust algorithm, we set the population size to be 50, and set other parameters to their default values[4].

Moreover, since the clustering results of *ccdByEnsemble* algorithm and *Squeezer* algorithm are fixed for a particular dataset when the parameters are fixed, only one run is used in the two algorithms. The GAClust algorithm is a genetic algorithm, so its outputs will differ in different runs. However, we observed in the experiments that the clustering error is very stable, so the clustering error of this algorithm is reported with its first run. In summary, we use one run to get the clustering errors for all the three algorithms.

5.3 Clustering Results on Congressional Voting (votes) Data

Fig. 2 shows the results on the *votes* dataset of different clustering algorithms. From Fig. 2, we can summarize the relative performance of these algorithms as follows Table 2.

Comparing to the *Squeezer* algorithm and the GAClust algorithm, the *ccdByEnsemble* algorithm performed best in 4 cases and second best in 4 cases. It never performed worst. And the average clustering error of the *ccdByEnsemble* algorithm was relatively smaller than that of other algorithms.

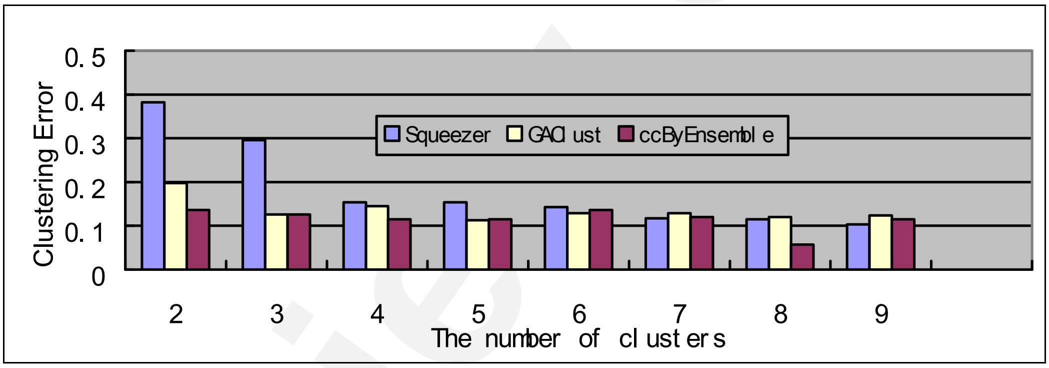

Fig.2. Clustering error vs. Different number of clusters (*votes* dataset)

Table 2: Relative performance of different clustering algorithms (*votes* dataset)

Ranking	1	2	3	Average Clustering Error
Squeezer	2	1	5	0.163
GAClust	3	2	3	0.136
ccdByEnsemble	4	4	0	**0.115**

Since in the integrated CE approach, *ccdByEnsemble*, we first run CSPA, HGPA and MCLA respectively, and selecting the one with the greatest ANMI as the final result. In this dataset, it is observed that CSPA has the greatest ANMI for 6 times, and MCLA has the greatest ANMI for 2 times. Hence, the reported results of *ccdByEnsemble* are dominated by CSPA.

[4] The source codes for GAClust are public available at: http://www.cs.umb.edu/~dana/GAClust/index.html. The readers may refer to this site for details about other parameters.

5.4 Clustering Results on Cancer Data

The experimental results on the *cancer* dataset are described in Fig. 3 and the summarization on the relative performance of the 3 algorithms is given in Table 3. From Fig. 3 and Table 3, although the average clustering accuracy of our algorithm is only a little better than that of the *Squeezer* and GAClust algorithm, while the cases of our algorithm that beat the other two algorithms are dominant in this experiment.

In this dataset, CSPA has the greatest ANMI for all cases and determine the clustering results of *ccdByEnsemble* absolutely. From this experiment and results reported in Section 5.3, it is clear that the clustering output of *ccdByEnsemble* is mainly determined by CSPA. That is, *ccdByEnsemble* can outperform the *Squeezer* and GAClust is mainly due to the effectiveness of CSPA.

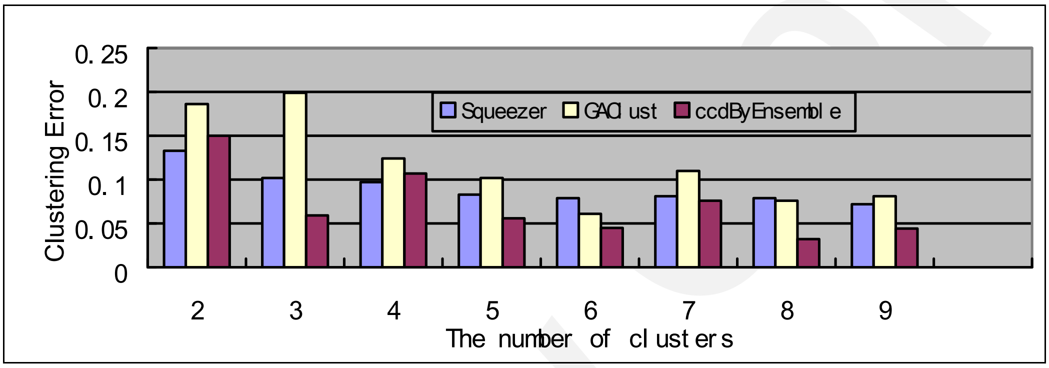

Fig.3. Clustering error vs. Different number of clusters (cancer dataset)

Table 3: Relative performance of different clustering algorithms (cancer dataset)

Ranking	1	2	3	Average Clustering Error
Squeezer	2	4	2	0.091
GAClust	0	2	6	0.117
ccdByEnsemble	6	2	0	**0.071**

5.5 Clustering Results on Mushroom Data

Because the mushroom dataset have 8124 records, CSPA failed to work on this larger dataset. So *ccdByEnsemble* uses only HGPA and MCLA in this experiment. That is, we first run HGPA and MCLA, and selecting the one with the greatest ANMI as the final result.

The experimental results on the *mushroom* dataset are described in Fig. 4 and Table 4. As Fig. 4 and Table 4 show, our algorithm and *Squezzer* algorithm outperform GAClust algorithm in this dataset. *Squezzer* algorithm achieves the best clustering performance. As we have argued in Section 5.4, the effectiveness of *ccdByEnsemble* mainly comes from CSPA. While CSPA failed to work in this larger dataset, which resulted in the unsatisfactory performance of *ccdByEnsemble* algorithm in this experiment. However, even in the absence of CSPA, *ccdByEnsemble* algorithm performed best in 2 cases.

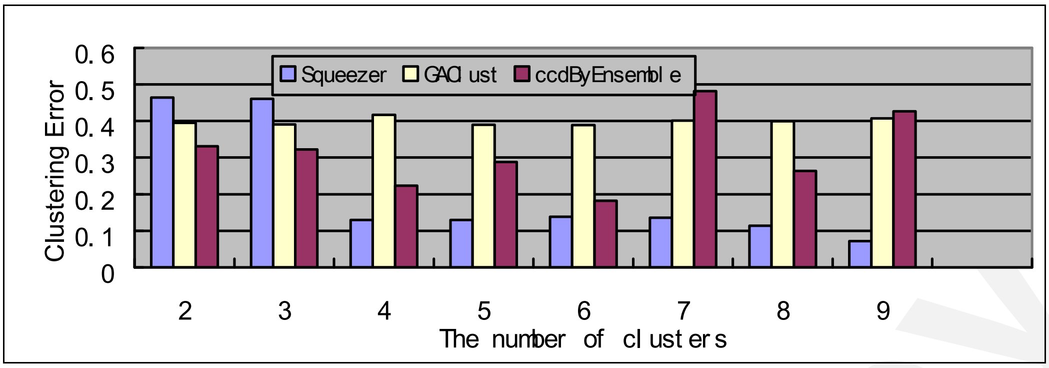

Fig.4. Clustering error vs. Different number of clusters (mushroom dataset)

Table 4: Relative performance of different clustering algorithms (mushroom dataset)

Ranking	1	2	3	Average Clustering Error
Squeezer	6	0	2	**0.206**
GAClust	0	4	4	0.393
ccdByEnsemble	2	2	2	0.315

5.6 Clustering Results on Zoo Data

The above *votes*, *cancer* and *mushroom* datasets have roughly balanced class distribution, which is very suitable for *ccdByEnsemble* algorithm because this algorithm desires to produce balanced clusters. In this Section, we test the performance of *ccdByEnsemble* algorithm on the Zoo dataset, which has unbalanced class distribution (See Table 5).

Table 5: Class Distribution of the Zoo Dataset

Class#	Set of animals
1	(41) aardvark, antelope, bear, boar, buffalo, calf, cavy, cheetah, deer, dolphin, elephant, fruitbat, giraffe, girl, goat, gorilla, hamster, hare, leopard, lion, lynx, mink, mole, mongoose, opossum, oryx, platypus, polecat, pony, porpoise, puma, pussycat, raccoon, reindeer, seal, sealion, squirrel, vampire, vole, wallaby,wolf
2	(20) chicken, crow, dove, duck, flamingo, gull, hawk, kiwi, lark, ostrich, parakeet, penguin, pheasant, rhea, skimmer, skua, sparrow, swan, vulture, wren
3	(5) pitviper, seasnake, slowworm, tortoise, tuatara
4	(13) bass, carp, catfish, chub, dogfish, haddock, herring, pike, piranha, seahorse, sole, stingray, tuna
5	(4) frog, frog, newt, toad
6	(8) flea, gnat, honeybee, housefly, ladybird, moth, termite, wasp
7	(10) clam, crab, crayfish, lobster, octopus, scorpion, seawasp, slug, starfish, worm

From Fig.5 and Table 6, we can see that the performance of *ccdByEnsemble* algorithm on the zoo dataset is not satisfactory compared with another two algorithms. It indicates that *ccdByEnsemble* with its current object function is not very suitable for datasets with unbalanced class distribution. However, it should be noted that the clustering performance of *ccdByEnsemble* is very close to that of the other two algorithms. That is, even in dataset with unbalanced class distribution, our algorithm can achieves comparative performance.

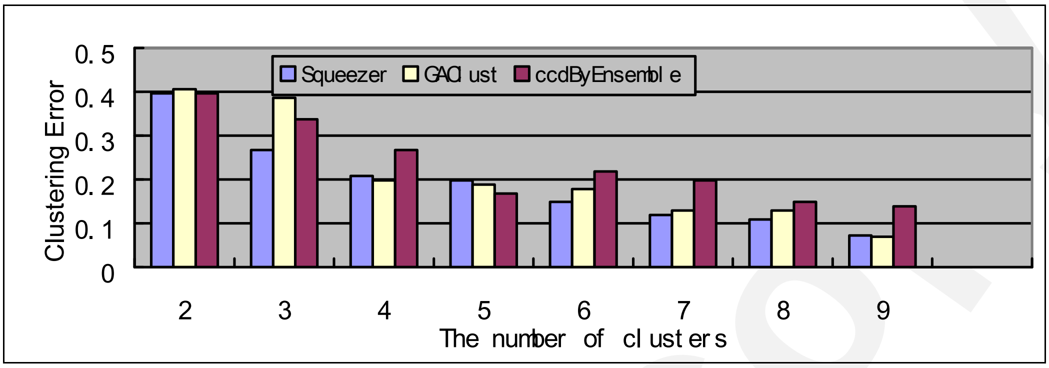

Fig.5. Clustering error vs. Different number of clusters (zoo dataset)

Table 6: Relative performance of different clustering algorithms (zoo dataset)

Ranking	**1**	**2**	**3**	**Average Clustering Error**
Squeezer	5	3	1	**0.190**
GAClust	2	4	2	0.210
ccdByEnsemble	2	1	5	0.234

5.7 Summary

The above experimental results on the four dataset demonstrate the effectiveness of cluster ensemble approach for clustering categorical dataset. One may argue that the results cannot precisely reflect that our method has better performance since our method only dominate on two datasets. However, from those results, we are confident to claim that our method could provide at least the same level of accuracy as other popular methods.

6. Conclusions

CE is a general knowledge reuse framework with many applications, and CDC is a special case in clustering research. Until recently, CDC and CE have been considered as separate research and application areas. Our main contribution in this paper is to explicitly state the equivalence between the CDC problem and CE problem from a *restricted* viewpoint for the first time, and point out that algorithms developed in both domains can be used interchangeably. Moreover, to verify our statement, we formally define the CDC problem as an optimization problem from the viewpoint of CE, and apply CE approach for clustering categorical data. Empirical evidences show that our idea is promising in practice.

For future work, we are planning to design *k*-means like clustering algorithms for categorical data that directly optimize the mutual information sharing based object function.

Acknowledgements

The comments and suggestions from the anonymous reviewers greatly improve the paper. We also like to express our thanks to Dr. Belur V. Dasarathy for his helpful suggestions on revising the paper. This work was supported by The High Technology Research and Development Program of China (Grant No. 2002AA413310, Grant No. 2003AA4Z2170, Grant No. 2003AA413021), the National Nature Science Foundation of China (Grant No. 40301038) and the IBM SUR Research Fund.

References

[1] A. Sehgal, U. B. Desai. 3D object recognition using Bayesian geometric hashing and pose clustering. Pattern Recognition, 2003, 36(3): 765-780.

[2] D. S. Boone, M. Roehm. Retail segmentation using artificial neural networks. Internal Journal of Research in Marketing, 2002, 19(3):287–301.

[3] V. Castelli, A. Thomasian, C-S Li. CSVD: Clustering and singular value decomposition for approximate similarity search in high-dimensional spaces. IEEE Transactions on Knowledge and Data Engineering, 2003, 15(3): 671-685.

[4] A. Popescul, G. W. Flake, S. Lawrence, L. H. Ungar, C. L. Giles. Clustering and identifying temporal trends in document databases. In: Proc of IEEE Advances in Digital Libraries 2000 (ADL 2000), 22-24 May 2000, Washington, DC, pp. 173-182.

[5] E.H. Han, G. Karypis, V. Kumar, B. Mobasher: Clustering based on association rule hypergraphs. In: SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, pp. 9-13, 1997.

[6] D. Gibson, J. Kleiberg, P. Raghavan. Clustering categorical data: an approach based on dynamic systems. In: Proc of VLDB’98, pp. 311-323, 1998.

[7] Y. Zhang, A. W. Fu, C. H. Cai, P. A. Heng. Clustering categorical data. In: Proc of ICDE’00, pp. 305-305, 2000.

[8] Z. Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In: SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, pp. 1-8, 1997.

[9] Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery, 1998, 2(3): 283-304.

[10] F. Jollois, M. Nadif. Clustering large categorical data. In: Proc of PAKDD’02, pp. 257-263, 2002

[11] Z. Huang, M. K. Ng. A fuzzy k-modes algorithm for clustering categorical data. IEEE Transaction on Fuzzy Systems, 1999, 7(4): 446-452.

[12] M. K. Ng, J. C. Wong. Clustering categorical data sets using tabu search techniques. Pattern Recognition, 2002, 35(12): 2783-2790.

[13] Y. Sun, Q. Zhu, Z. Chen. An iterative initial-points refinement algorithm for categorical data clustering. Pattern Recognition Letters, 2002, 23(7): 875-884.

[14] V. Ganti, J. Gehrke, R. Ramakrishnan. CACTUS-clustering categorical data using summaries. In: Proc of KDD’99, pp. 73-83, 1999.

[15] S. Guha, R. Rastogi, K. Shim. ROCK: a robust clustering algorithm for categorical attributes. In: Proc of ICDE'99, pp. 512-521, 1999.

[16] K. Wang, C. Xu, B. Liu. Clustering transactions using large items. In: Proc of CIKM'99, pp. 483-490, 1999.

[17] C. H. Yun, K. T. Chuang, M. S. Chen. An efficient clustering algorithm for market basket data based on small large ratios. In: Proc of COMPSAC'01, pp. 505-510, 2001.

[18] C. H. Yun, K. T. Chuang, M. S. Chen. Using category based adherence to cluster market-basket data. In: Proc of ICDM'02, pp.546-553, 2002.

[19] J. Xu, S. Y. Sung. Caucus-based transaction clustering. In: Proc of DASFAA'03, pp. 81-88, 2003.

[20] Z. He, X. Xu, S. Deng. Squeezer*:* an efficient algorithm for clustering categorical data. Journal of Computer Science & Technology, 2002,17(5): 611-624.

[21] D. Barbara, Y. Li, J. Couto. COOLCAT: an entropy-based algorithm for categorical clustering. In: Proc of CIKM'02, pp. 582-589, 2002.

[22] Y. Yang, S. Guan, J. You. CLOPE: a fast and effective clustering algorithm for transactional data. In: Proc of KDD'02, pp. 682-687, 2002.

[23] F. Giannotti, G. Gozzi, G. Manco. Clustering transactional data. In: Proc of PKDD'02, pp. 175-187, 2002.

[24] D. Cristofor, D. Simovici. Finding median partitions using information-theoretical-based genetic algorithms. Journal of Universal Computer Science, 2002, 8(2): 153-172.

[25] A. Strehl, J. Ghosh. Cluster Ensembles – A Knowledge Reuse Framework for Combining Partitions. In: Proc. of the 8th National Conference on Artificial Intelligence and 4th Conference on Innovative Applications of Artificial Intelligence, pp. 93-99, 2002.

[26] A. Strehl. Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining. PhD thesis, The University of Texas at Austin, May 2002.

[27] D. Frossyniotis, M. Pertselakis, A. Stafylopatis. A multi-clustering fusion algorithm. In: Proc. of the Second Hellenic Conference on AI, pp. 225-236, 2002.

[28] T. Qian, Y. S. Ching, Y. Tang. Sequential combination method for data clustering analysis. Journal of Computer Science & Technology, 2002, 17(2): 118-128.

[29] P-E. Jouve, N. Nicoloyannis. A method for aggregating partitions, applications in K.D.D. In: Proc of PAKDD'03, pp. 411-422, 2003.

[30] Y. Zeng, J. Tang, J. Garcia-Frias, G. Gao. An adaptive meta-clustering approach: combining the information from different clustering results. In: Proc of CSB'02, pp. 276-287, 2002.

[31] P-E. Jouve, N. Nicoloyannis. A new method for combining partitions, applications for cluster ensembles in KDD. In: Parallel and Distributed computing for Machine Learning Workshop, conjunction with ECML'03 and PKDD'03, pp. 35-46, 2003.

[32] G. Karpis, V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal of Scientific Computing, 1998, 20(1): 359-392.

[33] G. Karypis, R. Aggarwal, V. Kumar, S. Shekhar. Multilevel hypergraph partitioning: applications in VLSI domain. In: Proceedings of the Design and Automation Conference, pp. 526-529, 1997.

[34] C. J. Merz, P. Merphy. UCI Repository of Machine Learning Databases, 1996. (Http://www.ics.uci.edu/~mlearn/MLRRepository.html).

[35] A. Topchy, A.K. Jain, W.Punch. Combining multiple weak clusterings. In: Proc. of ICDM'03, pp. 331-338, 2003.

[36] A. Strehl, J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. Journal on Machine Learning Research, 2002, 3:583-617.