2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

PARTCAT: A Subspace Clustering Algorithm for High Dimensional Categorical Data

Guojun Gan, Jianhong Wu, and Zijiang Yang

***Abstract*— A new subspace clustering algorithm, PARTCAT, is proposed to cluster high dimensional categorical data. The architecture of PARTCAT is based on the recently developed neural network architecture PART, and a major modification is provided in order to deal with categorical attributes. PARTCAT requires less number of parameters than PART, and in particular, PARTCAT does not need the distance parameter that is needed in PART and is intimately related to the similarity in each fixed dimension. Some simulations using real data sets to show the performance of PARTCAT are provided.**

I. Introduction

Data clustering is an unsupervised process that divides a given data set into groups or clusters such that the points within the same cluster are more similar than points across different clusters. Data clustering is a primary tool of data mining, a process of exploration and analysis of large amount of data in order to discover useful information, thus has found applications in many areas such as text mining, pattern recognition, gene expressions, customer segmentations, image processing, to name just a few. An overview of the topic can be found in [1]–[4].

Although various algorithms have been developed, most of these clustering algorithms do not work efficiently for high dimensional data because of the inherent sparsity of data [5]. Consequently, dimension reduction techniques such as PCA (Principal Component Analysis) [6] and Karhunen-Loève Transformation [7], or feature selection techniques have been used in order to reduce the dimensionality before clustering. Unfortunately, such dimension reduction techniques require selecting certain dimensions in advance, which may lead to a significant loss of information. This can be illustrated by considering a 3-dimensional data set that has 3 clusters (See Fig. 1): one is embedded in (x, y)-plane, another is embedded in (y, z)-plane and the third one is embedded in (z, x)-plane. For such a data set, an application of a dimension reduction or a feature selection method is unable to recover all the cluster structures, because the 3 clusters are formed in different subspaces. In general, clustering algorithms based on dimension reduction or feature selection techniques generate clusters that may not fully reflect the original cluster structures. As a result, projected clustering or subspace clustering has been introduced to identify the clusters embedded in the subspaces of the original space [7], [8].

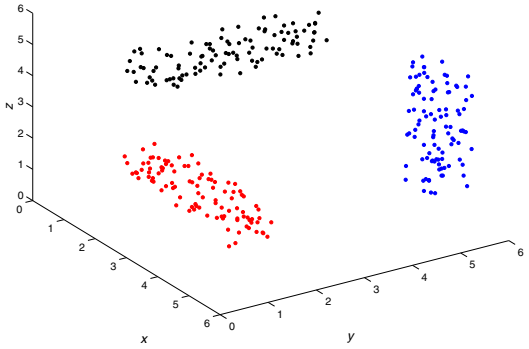

Fig. 1. A data set with three clusters embedded in different planes. The blue cluster is embedded in the (x, y)-plane, the red cluster is embedded in the (y, z)-plane, and the black cluster is embedded in the (z, x)-plane.

After a subspace clustering algorithm called CLIQUE was first introduced by Agrawal et. al. [7], several subspace clustering algorithms have been developed, such as PART [5], PROCLUS [8], ORCLUS [9], FINDIT [10], SUBCAD [11] and MAFIA [12]. However, working only on numerical data of these algorithms restricts their uses in data mining where categorical data sets are frequently encountered. In this paper, we propose an algorithm called PARTCAT (Projective Adaptive Resonance Theory for CATegorical data clustering) based on a neural network architecture PART (Projective Adaptive Resonance Theory) for clustering high dimensional categorical data.

PART [5], [13], [14] is a new neural network architecture that was proposed to find clusters embedded in subspaces of high dimensional spaces. The neural network architecture in PART is based on the well known ART (Adaptive Resonance Theory) developed by Carpenter and Grossberg [15]–[17] (See Fig. 2). In PART, a so-called selective output signaling mechanism is provided in order to deal with the inherent sparsity in the full space of the high dimensional data points. Under this selective output signaling mechanism, signal generated in a neural node in the input layer can be transmitted to a neural node in the clustering layer only when the signal is similar to the top-down weight between the two neural nodes. Thus with this selective output signaling mechanism, PART is able to find dimensions where subspace clusters can be found.

Guojun Gan is a PhD candidate at the Department of Mathematics and Statistics, York University, Toronto, ON, Canada, M3J 1P3 (email: gjgan@mathstat.yorku.ca).

Jianhong Wu is with the Department of Mathematics and Statistics, York University, Toronto, ON, Canada, M3J 1P3 (fax: 416-736-5757; email: wujh@mathstat.yorku.ca).

Zijiang Yang is with the Department of Mathematics and Statistics, York University, Toronto, ON, Canada, M3J 1P3 (fax: 416-736-5757; email: zyang@yorku.ca).

0-7803-9490-9/06/$20.00/©2006 IEEE

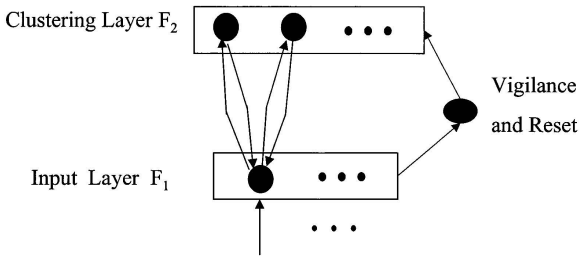

Fig. 2. Simplified configuration of ART architecture consisting of an input layer F_1, a clustering layer F_2 and a reset subsystem.[1]

The basic architecture of PART consists of three layers and a reset mechanism (See Fig. 3). The three layers are input and comparison layer(F_1 layer), clustering layer(F_2 layer) and a hidden layer associated with each F_1 layer neural node v_i for similarity check to determine whether the neural node v_i is active to a F_2 layer neural node v_j. PART Tree is an extension of the basic PART architecture. When all data points in the data set are clustered, we obtain a F_2 layer with projected clusters or outliers in each F_2 layer node, then data points in each projected cluster in F_2 layer nodes form a new data set, the same process is applied to each of those new data sets with a higher vigilance condition. This process is continued until some stop condition is satisfied, then the PART Tree is obtained.

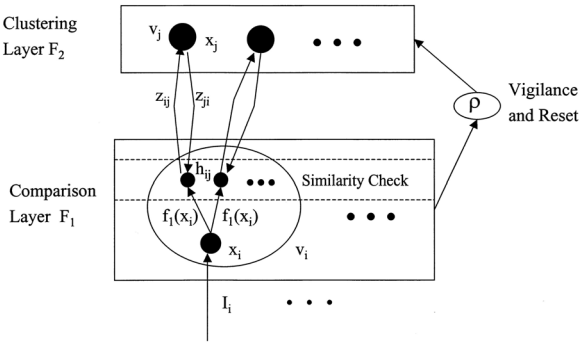

Fig. 3. PART architecture. F_1 layer is the input and comparison layer, F_2 layer is the clustering layer. In addition, there are a reset mechanism and a hidden layer associated with each F_1 node v_i for similarity check to determine whether v_i is active to an F_2 node v_j.[1]

PART is very effective to find the subspace in which a cluster embedded, but the difficulty of choosing some parameters in the algorithm of PART restricts its application. For example, it is very difficult for users to choose an appropriate value for σ, the distance parameter used to control the similarities between data points in PART. On the one hand, if we choose a small value for σ, the algorithm may not capture the similarities of two similar data points and may end up with each single data point as a cluster; on the other hand, if we choose a large value for σ, the algorithm may not differentiate two dissimilar data points and may produce a single cluster containing all data points.

The algorithm PARTCAT proposed in this paper follows the same neural architecture as PART. The principal difference between PARTCAT and PART is the up-bottom weight and the learning phase. In addition, the important feature of PARTCAT that σ is not needed is trivial for categorical data, since the distance between two categories takes one of two possible values: 0 if they are identical or 1 if they are different. The remaining part of this paper is organized as follows. In Section II, the PART algorithm is briefly reviewed. In Section III and Section IV, PARTCAT is introduced in detail. In Section V, experimental results on real data sets are presented to illustrate the performance of PARTCAT. In Section VI, some concluding remarks are given for PARTCAT.

II. Basic architecture of PART

The basic PART architecture consists of three components: input layer(comparison layer) F_1, clustering layer F_2 and a reset mechanism [5]. Let the nodes in F_1 layer be denoted by $v_i, i = 1, 2, ..., m$; nodes in F_2 layer be denoted by $v_j, j = m+1, ..., m+n$; the activation of an F_1 node v_i be denoted by x_i, the activation of an F_2 node v_j be denoted by x_j. Let the bottom-up weight from v_i to v_j be denoted by z_{ij}, the top-down weight from v_j to v_i be denoted by z_{ji}. Then in PART, the selective output signal of an F_1 node v_i to a committed F_2 node v_j is defined by

$$h_{ij} = h(x_i, z_{ij}, z_{ji}) = h_\sigma(f_1(x_i), z_{ji}) l(z_{ij}), \tag{1}$$

where f_1 is a signal function, $h_\sigma(\cdot, \cdot)$ is defined as

$$h_\sigma(a, b) = \begin{cases} 1, & \text{if } d(a, b) \le \sigma, \\ 0, & \text{if } d(a, b) > \sigma, \end{cases} \tag{2}$$

with $d(a, b)$ being a quasi-distance function, and $l(\cdot)$ is defined as

$$l(z_{ij}) = \begin{cases} 1, & \text{if } z_{ij} > \theta, \\ 0, & \text{if } z_{ij} \le \theta, \end{cases} \tag{3}$$

with θ being 0 or a small number to be specified as a threshold, σ is a distance parameter.

A F_1 node v_i is said to be active to v_j if $h_{ij} = 1$, and inactive to v_j is $h_{ij} = 0$.

In PART, a F_2 node v_j is said to be a winner either if $\Gamma \neq \phi$ and $T_j = \max \Gamma$, or if $\Gamma = \phi$ and node v_j is the next non-committed node in F_2 layer, where Γ is a set defined as $\Gamma = \{T_k : F_2 \text{ node } v_k \text{ is committed and has not been reset on the current trial}\}$ with T_k being defined as

$$T_k = \sum_{v_i \in F_1} z_{ik} h_{ik} = \sum_{v_i \in F_1} z_{ik} h(x_i, z_{ik}, z_{ki}). \tag{4}$$

A winning F_2 node will become active and all other F_2 nodes will become inactive, since F_2 layer makes a choice by winner-take-all paradigm:

$$f_2(x_j) = \begin{cases} 1, & \text{if node } v_j \text{ is a winner}, \\ 0, & \text{otherwise}. \end{cases} \tag{5}$$

[1]Reprinted from Neural Networks, Volume 15, Y. Cao and J. Wu, Projective ART for clustering data sets in high dimensional spaces, p106, Copyright (2002), with permission from Elsevier.

For the vigilance and reset mechanism of PART, if a winning (active) F_2 node v_j does not satisfy some vigilance conditions, it will be reset so that the node v_j will always be inactive during the rest of the current trial. The vigilance conditions in PART also control the degree of similarity of patterns grouped in the same cluster. For a winning F_2 node v_j, it will be reset if and only if

$$r_j < \rho, \tag{6}$$

where $\rho \in \{1, 2, ..., m\}$ is a vigilance parameter, and r_j is defined as

$$r_j = \sum_i h_{ij} \tag{7}$$

Therefore, the vigilance parameter ρ controls the size of subspace dimensions, and the distance parameter σ controls the degree of similarity in a specific dimension involved. For real world data, the distance parameter σ is difficult for user to choose, but in our algorithm PARTCAT, the distance parameter is no longer needed.

In PART, the learning are determined by the following formula. For the committed F_2 node v_j which has passed the vigilance test, the new bottom-up weight is defined as

$$z_{ij}^{new} = \begin{cases} \frac{L}{L-1+|X|}, & \text{if } F_1 \text{ node } v_i \text{ is active to } v_j, \\ 0, & \text{if } F_1 \text{ node } v_i \text{ is inactive to } v_j, \end{cases} \tag{8}$$

where L is a constant and $|X|$ denotes the number of elements in the set $X = \{i : h_{ij} = 1\}$, and the new top-down weight is defined as

$$z_{ji}^{new} = (1 - \alpha) z_{ji}^{old} + \alpha I_i, \tag{9}$$

where $0 \leq \alpha \leq 1$ is the learning rate.

For a non-committed winner v_j, and for every F_1 node v_i, the new weights are defined as

$$z_{ij}^{new} = \frac{L}{L - 1 + m}, \tag{10}$$

$$z_{ji}^{new} = I_i. \tag{11}$$

In PART, each committed F_2 node v_j represents a subspace cluster C_j. Let D_j be the set of subspace dimensions associated with C_j, then $i \in D_j$ if and only if $l(z_{ij}) = 1$, i.e. the set D_j is determined by $l(z_{ij})$.

III. Basic architecture of PARTCAT

PARTCAT is an extension of PART, which has the same neural network architecture as PART. We use the same notations defined as in Section II. In PARTCAT, the selective output signal from F_1 node v_i to F_2 node v_j is defined as

$$h_{ij} = h(x_i, z_{ij}, z_{ji}) = \delta(x_i, z_{ji}) l(z_{ij}), \tag{12}$$

where $l(z_{ij})$ is defined as in Equation (3), and $\delta(\cdot, \cdot)$ is the Simply Matching distance [18], i.e.

$$\delta(a, b) = \begin{cases} 1, & \text{if } a = b, \\ 0, & \text{if } a \neq b, \end{cases} \tag{13}$$

The learning formula of PARTCAT have little difference from that of PART. For a non-committed winner v_j and for every F_1 node v_i, the new weights are defined in Equations (10) and (11). For the committed winning F_2 node v_j which has passed the vigilance test, the bottom-up weight is updated based on the formula defined in Equation (8). Nevertheless, the top-down weight is updated according to the following rule.

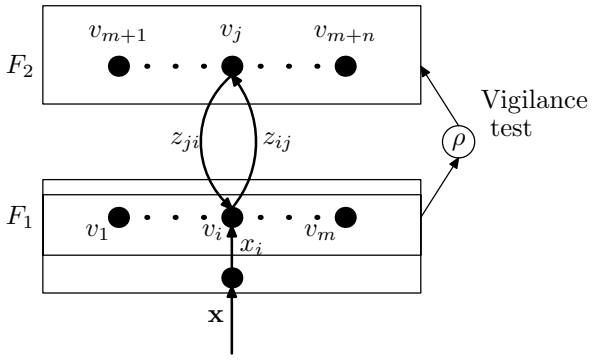

Fig. 4. PARTCAT architecture: F_1 layer is the input and comparison layer, F_2 layer is the clustering layer. In addition, there are a reset mechanism and a hidden layer associated with each node v_i in F_1 layer for similarity check to determine whether v_i is actively relative to node v_j in F_2 layer.

For the learning rule of top-down weight z_{ji}^{new} of the committed winning F_2 node v_j which has passed the vigilance test, we need to change it so that it is suitable for categorical values. To do this, let T_s be a symbol table of the input data set and $T_f(C_j)$ be the frequency table for F_2 node v_j(See Appendix), where C_j is the cluster in node v_j. Let $f_{kr}(C_j)$ be the number of elements in cluster C_j whose kth attribute takes value A_{kr}, i.e.

$$f_{kr}(C_j) = |\{\mathbf{x} \in C_j : x_k = A_{kr}\}|,$$

where x_k is the kth component of $\mathbf{x}$ and A_{kr} is a state of the kth variable's domain $DOM(A_k) = (A_{k1}, A_{k2}, ..., A_{kn_k})$.

Let N_j denote the number of elements in F_2 node v_j, then we have

$$N_j = \sum_{r=1}^{n_k} f_{kr}(C_j), \quad \forall k, j.$$

Now we can define the learning rule for top-down weight z_{ji}^{new} of the committed F_2 node v_j to v_i as

$$z_{ji}^{new} = A_{il_i}, i = 1, 2, ..., m,$$

where $l_i (1 \leq l_i \leq n_i)$ is defined by

$$l_i = \arg \max_{1 \leq r \leq n_i} f_{ir}^{new}(C_j).$$

where $f_{ir}^{new}(C_j)$ is defined as

$$f_{ir}^{new}(C_j) = f_{ir}^{old}(C_j) + \delta(I_i, A_{ir}).$$

where $\delta(\cdot, \cdot)$ is the Simple Matching distance defined in Equation (13).

Therefore, in PARTCAT, we no longer need the distance parameter σ since we use Simple Matching distance in our algorithm.

A. PARTCAT tree

PARTCAT tree is an extension of the basic PARTCAT architecture. Data points in a F_2 node can be further clustered by increasing the vigilance parameter ρ. When each F_2 node is clustered using the basic PARTCAT architecture, we obtain a new F_2 layer consisting of new sets of projected clusters. A PARTCAT tree is obtained when this process is continued for the new F_2 layer. A natural stop condition for this process is $\rho > m$. Another stop condition is that the number of elements in a F_2 layer is less than N_{min}, where N_{min} is a constant, i.e. when a F_2 node has less than N_{min} elements, the cluster in this node will not be further clustered.

IV. Algorithms

The principle difference between the algorithm PARTCAT and the algorithm PART is the learning rule for top-down weight of the committed winning F_2 node that has passed the vigilance test.

A. F_1 activation and computation of h_{ij}

Let I_i be the input from the ith node of F_1 layer, then we compute h_{ij} by Equation (12) with $x_i = I_i$, i.e.

$$h_{ij} = \delta(I_i, z_{ji}) l(z_{ij}). \tag{14}$$

where $\delta(\cdot, \cdot)$ is the Simple Matching distance.

B. F_2 activation and selection of winner

For the committed F_2 nodes, we compute T_j in Equation (4), therefore by Equation (14), we have

$$T_j = \sum_{v_i \in F_1} z_{ij} h_{ij} = \sum_{v_i \in F_1} z_{ij} \delta(I_i, z_{ji}) l(z_{ij}). \tag{15}$$

To select the winner, we use the same rule as PART. Let $\Gamma = \{T_j : F_2 \text{ node } v_j \text{ is committed and has not been reset on the current trial}\}$, then a F_2 node v_j is said to be a winner either if $\Gamma \neq \phi$ and $T_j = \max \Gamma$, or if $\Gamma = \phi$ and node v_j is the next non-committed node in F_2 layer.

C. Vigilance test

PARTCAT uses the same reset mechanism as PART. If a winner can not pass a vigilance test, it will be reset by the reset mechanism. A winning committed F_2 node v_j will be reset if and only if

$$\sum_i h_{ij} < \rho. \tag{16}$$

where $\rho \in \{1, 2, ..., m\}$ is a vigilance parameter.

D. Learning and Dimensions of projected clusters

The learning scheme in PARTCAT is different from PART. We specify the learning formula of PARTCAT in Section III. Each committed F_2 node v_j represents a projected cluster C_j, let D_j be the corresponding set of subspace dimensions associated with C_j, then $i \in D_j$ if and only if $l(z_{ij}) = 1$. The pseudo-code of the PARTCAT tree algorithm is described in Algorithm 1.

Algorithm 1 The tree algorithm of PARTCAT.

Require: D - the categorical data set,m - number of dimensions, k - number of clusters;

Number of m nodes in F_1 layer $\Leftarrow$ number of dimensions;
number of k nodes in F_2 layer $\Leftarrow$ expected maximum number of clusters that can be formed at each clustering level;
Set values for L, ρ_0, ρ_h and θ;
$\rho \Leftarrow \rho_0$;
$S \Leftarrow D$;
repeat
  Set all F_2 nodes as being non-committed;
  repeat
    for all data points in S **do**
      Compute h_{ij} for all F_1 nodes v_i and F_2 nodes v_j;
      if at least one of F_1 nodes are committed **then**
        Compute T_j for all committed F_2 nodes v_j;
      end if
      repeat
        Select the winning F_2 node v_J;
        if the winner is a committed node **then**
          Compute r_J;
          if $r_J < \rho$ **then**
            Reset the winner v_J;
          end if
        else
          break;
        end if
      until $r_j \geq \rho$
      if no F_2 node can be selected **then**
        put the input data into outlier O;
        break;
      end if
      Set the winner v_J as the committed, and update the bottom-up and top-down weights for winner node v_j;
    end for
  until the difference of the output of the clusters in two successive step becomes sufficiently small
  for all cluster C_j in F_2 layer **do**
    Compute the associated dimension set D_j, then set $S \Leftarrow C_j$ and $\rho \Leftarrow \rho + \rho_h$ and do the same process;
  end for
  For the outlier set O, set $S \Leftarrow O$ and do the same process;
until some stop criterion is satisfied

V. Experimental Evaluations

PARTCAT is coded in C++ programming language. Experiments on real data sets are conducted on a Sun Blade 1000 workstation. For all the simulations in this section, we specify $L = 2$ and $\theta = 0$. Some parameters (e.g. the number of dimensions m and the size of the data set n) are determined by the specific data set. Other parameters (e.g. the initial subspace dimensions ρ_0 and the dimension step ρ_h) are chosen by experiments for the individual data set.

For the purpose of comparison of clustering results, we also implement the k-Modes algorithm [19] which is well-know for clustering categorical data sets. We apply both algorithms to the data sets. In the k-Modes algorithm, we choose the initial modes randomly from the data set.

A. Data sets

Instead of generating synthetic data to validate the clustering algorithm, we use real data sets to test our algorithm for two reasons. Firstly, synthetic data sets may not well represent real world data [19]. Secondly, most of the data generation methods were developed for generating numeric data. The real data sets used to test PARTCAT are obtained from UCI Machine Learning Repository [20]. All these data sets have class labels assigned to the instances.

1) Soybean data: The first real data set is the soybean disease data set [20], obtained from the UCI Machine Learning Repository. The soybean data set has 47 records each of which is described by 35 attributes. Each record is labeled as one of the 4 diseases: diaporthe stem rot, charcoal rot, rhizoctonia root rot and phytophthora rot. Except for the phytophthora rot which has 17 instances, all other diseases have 10 instances each. We use D1, D2,D3,D4 to denote the 4 diseases. Since there are 14 attributes that have only one category, we only selected other 21 attributes for the purpose of clustering.

2) Zoo data: The second real data set is the zoo data [20]. The zoo data has 101 instances each of which is described by 18 attributes. Since one of the attributes is animal name which is unique for each instance and one of the attributes is animal type which can be treated as class label, we only selected other 16 attributes for the purpose of clustering. Each instance in this data set is labeled to be one of 7 classes.

3) Mushroom data: The third real data set is the mushroom data set [20], also obtained from UCI Machine Learning Repository. There are total 8124 records each of which is described by 22 attributes in the data set, and all attributes of this data set are nominally valued. There are some missing values in this data set, since the missing value can be treated as a special state of the categorical variable, we use the whole data set for clustering. Each instance in the data set is labeled to be one of the two classes: edible(e) and poisonous(p).

B. Results

For the soybean data set, we set number of initial subspace dimensions ρ_0 to be 7, dimension step ρ_h to be 1, expected maximum number of clusters to be formed at each clustering level k to be 4, and the minimum members for each cluster N_{min} to be 20. Apply PARTCAT to the soybean data set using these parameters, we got the results described in Table I. Table II gives the results of applying the k-Modes algorithm to the soybean data set.

TABLE I

PARTCAT: The misclassification matrix of the soybean data set.

	D1	D2	D3	D4
C1	10	0	1	0
C2	0	10	0	0
C3	0	0	9	0
C4	0	0	0	17

TABLE II

k-Modes: The misclassification matrix of the soybean data set.

	D1	D2	D3	D4
C1	7	0	0	10
C2	2	10	0	0
C3	1	0	0	7
C4	0	0	10	0

For each cluster of the soybean data set, the subspace dimensions associated with each cluster found by PARTCAT is described in Table III. From Table III, we see that the subspace dimensions associated with different clusters are different.

Also from Table I and Table II, we see that there is only one point which is misclassified by the PARTCAT algorithm, but there are ten points which are misclassified by the k-Modes algorithm.

TABLE III

PARTCAT: The subspace dimensions associated with each cluster of the soybean data set.

Clusters	Subspace dimensions
C1	2, 3, 11, 16, 18, 19, 21
C2	2, 3, 8, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21
C3	3, 7, 15, 18, 19, 20
C4	2, 7, 11,14, 15, 17, 18, 19, 20, 21

To cluster the zoo data using PARTCAT, we set number of initial subspace dimensions ρ_0 to be 3, dimension step ρ_h to be 1, expected maximum number of clusters to be formed at each clustering level k to be 5, and the minimum members for each cluster N_{min} to be 20. Apply PARTCAT to this data set using these parameters, we got the results described in Table IV. Table V shown the results produced by the k-Modes algorithm.

For the 7 clusters of the zoo data, the subspace information for each cluster is described in Table VI. Compare the results in Table IV and the results in Table V, we see that the k-Modes algorithm may split a large cluster in the whole data space.

For the mushroom data set, we set number of initial subspace dimensions ρ_0 to be 8, dimension step ρ_h to be

TABLE IV

PARTCAT: THE MISCLASSIFICATION MATRIX OF THE ZOO DATA.

	1	2	3	4	5	6	7
C1	35	0	0	0	0	0	0
C2	4	0	0	0	0	0	0
C3	0	4	3	0	2	0	1
C4	2	16	0	0	0	0	0
C5	0	0	0	0	2	8	8
C6	0	0	2	0	0	0	1
C7	0	0	0	13	0	0	0

TABLE V

k-MODES: THE MISCLASSIFICATION MATRIX OF THE ZOO DATA.

	1	2	3	4	5	6	7
C1	22	0	0	0	0	0	0
C2	0	0	0	0	0	8	10
C3	0	11	0	0	0	0	0
C4	0	0	4	0	4	0	0
C5	19	0	0	0	0	0	0
C6	0	0	1	13	0	0	0
C7	0	9	0	0	0	0	0

2, expected maximum number of clusters to be formed at each clustering level k to be 4, and the minimum members for each cluster N_{min} to be 500. If we apply PARTCAT to the mushroom data using these parameters values, we got results described in Table VII.

If we cluster the mushroom data into 20 clusters using PARTCAT, we see that most of the clusters are correct, except 6 clusters that contain both edible and poisonous mushrooms. Also we got some big clusters, such as clusters C4 and C12 which have more that 1000 data points. The subspace information of these clusters is described in Table IX. This is another example show that subspaces in which different

TABLE VI

PARTCAT: THE SUBSPACE DIMENSIONS ASSOCIATED WITH EACH CLUSTER OF THE ZOO DATA SET.

Clusters	Subspace dimensions
C1	1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13
C2	2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 15, 16
C3	1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 15
C4	5, 9, 10, 11, 12, 13, 14
C5	2, 3, 4, 12, 14, 16
C6	1, 2, 4, 5, 7, 11, 12, 14, 15, 16
C7	1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 13, 14

TABLE VII

PARTCAT: THE MISCLASSIFICATION MATRIX OF THE MUSHROOM DATA.

Clusters	e	p	Clusters	e	p
C1	64	0	C11	32	72
C2	448	216	C12	0	1296
C3	0	256	C13	192	0
C4	1920	0	C14	0	270
C5	96	96	C15	0	432
C6	288	0	C16	0	270
C7	0	104	C17	0	120
C8	0	414	C18	0	222
C9	384	72	C19	304	36
C10	192	0	C20	288	40

TABLE VIII

k-MODES: THE MISCLASSIFICATION MATRIX OF THE MUSHROOM DATA.

Clusters	e	p	Clusters	e	p
C1	0	333	C11	0	136
C2	623	16	C12	0	224
C3	0	774	C13	15	170
C4	576	303	C14	70	20
C5	0	258	C15	793	0
C6	0	241	C16	59	222
C7	0	550	C17	0	41
C8	694	259	C18	7	179
C9	0	168	C19	302	1
C10	362	0	C20	707	21

clusters are embedded are different.

If we cluster the mushroom data into 20 clusters using the k-Modes algorithm, the number of correct clusters is less than the number of correct clusters produced by the PARTCAT algorithm. Also the size of the largest cluster obtained by the k-Modes algorithm is much smaller than the size of the largest cluster obtained by the PARTCAT algorithm.

TABLE IX

PARTCAT: THE SUBSPACE DIMENSIONS ASSOCIATED WITH EACH CLUSTER OF THE MUSHROOM DATA SET.

Clusters	Subspace dimensions
C1	1, 4, 6, 7, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19
C2	4,6, 7, 10, 12,14, 15, 16, 17, 18, 19
C3	4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19
C4	5, 6, 16, 17, 18
C5	6, 7, 11, 12, 13, 14, 15, 16, 17, 18, 19, 22
C6	6, 7, 10, 12, 13, 14, 15, 16, 17, 18
C7	6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19
C8	4, 6, 7, 8, 9, 10, 11, 12, 16, 17, 18, 19, 20, 21
C9	6, 8, 12, 14, 15, 16, 17, 18
C10	1, 4, 5, 6, 7, 8, 10, 11, 12, 14, 15, 16, 17, 18, 19, 22
C11	4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19
C12	4, 5, 6, 7, 8, 10, 11, 12, 13, 16, 17, 18, 19, 20
C13	4, 5, 6, 7, 8, 10, 11, 12, 13,16, 17, 18, 19, 20, 21, 22
C14	1, 2, 4, 6, 7, 8, 9, 10, 11, 16, 17, 18, 19, 20, 21
C15	2, 4, 6, 7, 8, 9, 10, 11, 12, 16, 17, 18, 19, 20, 21
C16	1, 2, 4, 6, 7, 8, 9, 10, 11, 16, 17, 18, 19, 20, 21
C17	4, 6, 7, 8, 9, 10, 11, 14, 16, 17, 18, 19, 20, 21, 22
C18	1, 2, 4, 6, 7, 8, 9, 10, 11, 16, 17, 18, 19, 20, 21
C19	4, 8, 10, 16, 17, 20
C20	4, 5, 10, 16, 18

VI. DISCUSSION AND CONCLUSIONS

Most traditional clustering algorithms do not work efficiently for high dimensional data. Due to the inherent sparsity of the data points, it is not feasible to identify interesting clusters in the whole data space. In order to deal with high dimensional data, some techniques such as feature selection have been applied before clustering. But these techniques require pruning off variables in advance which may lead to unreliable clustering results. High dimensional categorical data have the same problem.

We propose a neural network clustering algorithm based on the neural network algorithm PART [5], [13], [14] in order to identify clusters embedded in the subspaces of the

data space instead of the whole data space. Unlike the PART algorithm, our algorithm PARTCAT is designed for clustering high dimensional categorical data.

Some subspace clustering algorithms such as CLIQUE [7], PROCLUS [8] and MAFIA [12] have been developed and studied, but they can only be applied to numerical data. To compare the clustering results with traditional clustering algorithms, we implement the k-Modes algorithm [19]. From the simulations described in Section V, we have seen that PARTCAT is able to generates better clustering results than the k-Modes algorithm. The reason for this is that PARTCAT identifies the clusters in the subspace of the whole data space, while the k-Modes algorithm finds the clusters in the whole data space. However, PARTCAT does not outperform SUBCAD, a subspace clustering algorithm proposed by Gan and Wu [11] for clustering high dimensional categorical data sets.

APPENDIX
SYMBOL TABLE AND FREQUENCY TABLE

The concepts of symbol table and frequency table are specific for categorical data sets. Given a d-dimensional categorical data set D, let A_j be the categorical variable of the jth dimension($1 \leq j \leq d$). We define its domain by $DOM(A_j) = \{A_{j1}, A_{j2}, ..., A_{jn_j}\}$ and we call $A_{jr}(1 \leq r \leq n_j)$ a state of the categorical variable A_j. Then a symbol table T_s of the data set is defined as:

$$T_s = (\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_d).$$

where $\mathbf{s}_j$ is a vector defined as $\mathbf{s}_j = (A_{j1}, A_{j2}, ..., A_{jn_j})^T$. Note that the symbol table for a data set is not unique, since states may have many permutations.

The frequency table of a cluster is computed according to a symbol table of that data set and it has exactly the same dimension as the symbol table. Let C be a cluster, then the frequency table $T_f(C)$ of cluster C is defined as

$$T_f(C) = (\mathbf{f}_1(C), \mathbf{f}_2(C), ..., \mathbf{f}_d(C)), \tag{17}$$

where $\mathbf{f}_j(C)$ is a vector defined as

$$\mathbf{f}_j(C) = (f_{j1}(C), f_{j2}(C), ..., f_{jn_j}(C))^T, \tag{18}$$

where $f_{jr}(C)(1 \leq j \leq d, 1 \leq r \leq n_j)$ is the number of data points in cluster C which take value A_{jr} at the jth dimension, i.e.

$$f_{jr}(C) = |\{\mathbf{x} \in C : x_j = A_{jr}\}|, \tag{19}$$

where x_j is the jth component of $\mathbf{x}$. For a given symbol table of the data set, the frequency table of each cluster is unique according to that symbol table.

ACKNOWLEDGMENT

We would like to thank Y. Cao for providing us two figures from his paper. We also would like to thank two anonymous referees for their invaluable comments and suggestions about the preliminary version of this work. The work of J. Wu was partially supported by Natural Sciences and Engineering Research Council of Canada and by Canada Research Chairs Program. The work of Z. Yang is partially supported by Faculty of Arts Research Grant, York University.

REFERENCES

[1] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, September 1999.

[2] A. Gordon, "A review of hierarchical classification," *Journal of the Royal Statistical Society. Series A (General)*, vol. 150, no. 2, pp. 119–137, 1987.

[3] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *The Computer Journal*, vol. 26, no. 4, pp. 354–359, November 1983.

[4] R. Cormack, "A review of classification," *Journal of the Royal Statistical Society. Series A (General)*, vol. 134, no. 3, pp. 321–367, 1971.

[5] Y. Cao and J. Wu, "Projective ART for clustering data sets in high dimensional spaces," *Neural Networks*, vol. 15, no. 1, pp. 105–120, January 2002.

[6] K. Yeung and W. Ruzzo, "Principal component analysis for clustering gene expression data," *Bioinformatics*, vol. 17, no. 9, pp. 763–774, September 2001.

[7] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," in *SIGMOD Record ACM Special Interest Group on Management of Data*, 1998, pp. 94–105.

[8] C. Aggarwal, J. Wolf, P. Yu, C. Procopiuc, and J. Park, "Fast algorithms for projected clustering," in *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*. ACM Press, 1999, pp. 61–72.

[9] C. Aggarwal and P. Yu, "Finding generalized projected clusters in high dimensional spaces," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*, W. Chen, J. F. Naughton, and P. A. Bernstein, Eds., vol. 29. ACM, 2000, pp. 70–81.

[10] K. Woo and J. Lee, "FINDIT: a fast and intelligent subspace clustering algorithm using dimension voting," Ph.D. dissertation, Korea Advanced Institue of Science and Technology, Department of Electrical Engineering and Computer Science, 2002.

[11] G. Gan and J. Wu, "Subspace clustering for high dimensional categorical data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 2, pp. 87–94, December 2004.

[12] H. Nagesh, S. Goil, and A. Choudhary, "A scalable parallel subspace clustering algorithm for massive data sets," in *2000 International Conference on Parallel Processing (ICPP'00)*. Washington - Brussels - Tokyo: IEEE, August 2000, pp. 477–486.

[13] Y. Cao, "Neural networks for clustering: Theory, architecture, algorithms and neural dynamics," Ph.D. dissertation, Department of Mathematics and Statistics, York University, Toronto, ON, Canada, October 2002.

[14] Y. Cao and J. Wu, "Dynamics of projective adaptive resonance theory model: the foundation of PART algorithm," *IEEE Transactions on Neural Networks*, vol. 15, no. 2, pp. 245–260, 2004.

[15] G. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition mathine," *Computer Vision, Graphics and Image Processing*, vol. 37, pp. 54–115, 1987.

[16] ——, "ART2: Self-organization of stable category recognition codes for analog input patterns," *Applied Optics*, vol. 26, pp. 4919–4930, 1987.

[17] ——, "ART3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures," *Neural Networks*, vol. 3, pp. 129–152, 1990.

[18] M. Anderberg, *Cluster Analysis for Applications*. New York: Academic Press, 1973.

[19] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data Mining and Knowledge Discovery*, vol. 2, no. 3, pp. 283–304, 1998.

[20] C. Blake and C. Merz, "UCI repository of machine learning databases," 1998, http://www.ics.uci.edu/~mlearn/MLRepository.html.