Generalizing Self-Organizing Map for Categorical Data

Chung-Chian Hsu

***Abstract*—The self-organizing map (SOM) is an unsupervised neural network which projects high-dimensional data onto a low-dimensional grid and visually reveals the topological order of the original data. Self-organizing maps have been successfully applied to many fields, including engineering and business domains. However, the conventional SOM training algorithm handles only numeric data. Categorical data are usually converted to a set of binary data before training of an SOM takes place. If a simple transformation scheme is adopted, the similarity information embedded between categorical values may be lost. Consequently, the trained SOM is unable to reflect the correct topological order. This paper proposes a generalized self-organizing map model that offers an intuitive method of specifying the similarity between categorical values via distance hierarchies and, hence, enables the direct process of categorical values during training. In fact, distance hierarchy unifies the distance computation of both numeric and categorical values. The unification is done by mapping the values to distance hierarchies and then measuring the distance in the hierarchies. Experiments on synthetic and real datasets were conducted, and the results demonstrated the effectiveness of the generalized SOM model.**

***Index Terms*—Categorical data, cluster analysis, distance hierarchy, neural networks, self-organizing map (SOM).**

I. Introduction

The self-organizing map (SOM), proposed by Kohonen, is an unsupervised neural network which projects high-dimensional data onto a low-dimensional grid [1] and [2]. The projected data preserves the topological relationship of the original data; therefore, this ordered grid can be used as a convenient visualization surface for showing various features of the training data, for example, cluster structures [3].

Since it was originally proposed, the SOM has had many applications. The applications initially focused on engineering, including image processing [4]–[7], process monitoring and control [8], [9], speech recognition [10], [11], and flaw detection in machinery [12]. Recently, applications to other fields have emerged including business and management, such as information retrieval [13], [14], medical diagnosis [15], [16], time-series prediction [17], [18], optimization [19], as well as financial forecasting and management [20]–[22].

Training an SOM using a dataset involves two key steps: Determining the best matching unit (BMU) and updating the BMU and its neighbors. The conventional SOM training algorithm can process only numeric data since determining the BMU from map units and updating BMU's topological neighbors are both based on a numeric distance function, typically the Euclidean. However, various types of huge data, including numeric, categorical, and even mixed data, are continuously accumulating due to the advent of information technology and the popularity of information systems. For instance, for student data in a campus database, the department attribute is categorical. For sales transactions in a sales database, the product attribute is categorical while the sales-amount attribute is numeric. Unable to process categorical data would confine the applicability of SOM.

A typical approach to processing categorical values in the current SOM training process is resorted to a preprocess such as binary encoding, which transforms each categorical attribute into a set of binary attributes in the way that each distinct categorical value is associated with one of the binary attributes. Consequently, after the transformation, all categorical attributes become binary attributes, which can thus be treated as numeric attributes with the domain of $\{0, 1\}$.

This straightforward approach has several drawbacks including increased dimensionality of the transformed relation, difficulty in maintaining the transformed relation schema, and inability to convey the semantics of the original attribute. Most importantly, this approach fails to preserve the similarity information embedded between categorical values. As a result, a trained SOM is unable to generate correct topological order when categorical values are involved and similar to one another in a different extent.

This paper aims at improving the ability of the SOM to process categorical data and mixed data in a direct and natural manner. We propose a generalized SOM (or GSOM) model such that the extended SOM can handle mixed data directly and reflect their topological relationship faithfully.

The reason why categorical values cannot be handled straightly in the conventional SOM algorithm is due to the lack of a direct representation and computation scheme of the distance between categorical values. To solve the problem, we propose a general distance representation structure, distance hierarchy, and then integrate it with the SOM to facilitate distance computation. In the integrated GSOM model, every attribute of the training dataset and its correspondent component of a GSOM unit both associate with a distance hierarchy . To compute the distance between a training pattern and a GSOM unit, we map the attribute values of the pattern and the correspondent components of the GSOM unit to their associated distance hierarchies. Then, the distance is computed by aggregating the distances between the mapping points in their hierarchies. Accordingly, our GSOM model makes possible the measuring

Manuscript received August 16, 2004; revised April 11, 2005.

The author is with the Department of Information Management, National Yunlin University of Science and Technology, Yunlin 640, Taiwan (e-mail: hsucc@yuntech.edu.tw).

Digital Object Identifier 10.1109/TNN.2005.863415

of the distance involving categorical values in a numeric way naturally.

It is worth mentioning that the distance computation scheme based on distance hierarchy unifies several traditional approaches including simple matching, binary encoding, and numeric value subtraction. The details are given in Section III-D.

A sophisticated transformation scheme that integrates similarity information between categorical values into numeric codes is possible to reflect the relationship of categorical data, for example, the one of the topological mappings of animals in [23], in which each animal is attributed with 13 binary qualities including *Size_small*, *Size_medium*, *Size_big*, *2_legs*, etc. For the encoding of an animal (for example, a hen), the three attributes *Size_small*, *2_legs*, and *Has_feathers* are set to 1, and the other attributes are set to 0. In addition, there have been many fuzzy-SOM hybrid realizations recently [24]–[29]. An alternative is resorting to fuzzy system concepts that can be used to convert categorical data into fuzzy membership values, which are numeric. However, these sophisticated schemes require extensive expert domain knowledge to specify the mapping between categorical values and their numeric codes. In complex cases, it might not be easy to devise an appropriate mapping. On the contrary, our proposed model via distance hierarchy offers an easy, intuitive approach to specifying the domain knowledge. Additionally, GSOM treats categorical and numeric values in a unified representation scheme, and its training algorithm processes categorical and numeric values directly.

This paper is organized as follows. In Section II, the basics of the SOM are briefly reviewed, and a detailed discussion of the process by which the conventional SOM handles categorical data is presented. In Section III, we propose a generalized SOM model which can directly handle categorical as well as mixed data. Section IV presents the experimental results of training GSOM using synthetic and real datasets. Comparisons to the conventional SOM are also reported. Section V gives concluding remarks.

II. Self-Organizing Map

A. The SOM Training

The SOM can nonlinearly project high-dimensional data onto a low-dimensional grid [4]. The projection preserves the topological order in the input space; hence, similar data patterns in the input space will be assigned to the same map unit or nearby units on the trained map. The core process of the projection is first, for each input pattern, determining its BMU from the map units. The BMU is the unit that is most similar to the input pattern. Then the process proceeds to update the BMU and its neighborhood units so as to reduce the difference between those units and the input pattern. In short, the two key steps in the SOM training algorithm are: 1) determining the BMU, and 2) updating the BMU and its neighbors.

Specifically, an input pattern is a high-dimensional vector of real numbers, $x = [x_1\ x_2 \dots x_n] \in \Re^n$, in the Euclidean space where x_i is the value of the ith component. Each unit on an associated SOM for an n-dimensional training dataset is also an n-dimensional real vector, $m_i = [m_{i1}\ m_{i2} \dots m_{in}] \in \Re^n$ where m_{ik} is the value of the kth component. The method for determining the BMU with respect to an input pattern x is to identify the unit that is most similar to x. A distance function is usually employed to measure the similarity. The smaller the distance is, the more similar they are. Formally, the BMU of an input x is defined as $m_v = \arg\min_i\{d(x, m_i)\}$ where m_i is a unit on the map. A typical method for computing the distance $d(x, m_i)$ is to use the Euclidian distance function

$$d(x, m_i) = \|x - m_i\| = \left(\sum_{k=1,n} (x_k - m_{ik})^2\right)^{1/2}. \quad (1)$$

Once the BMU is identified, the BMU and its neighbors are to be updated to reduce the difference with the input pattern. The updating is centered at the BMU, and the adjustment amount decreases with the increasing distance to the BMU. Similarly, the update neighborhood also decreases with the increasing training epochs. Formally, the update rule for a neighborhood unit m_i is

$$m_i(t+1) = m_i(t) + \alpha(t)h_{j,i}(t)[x(t) - m_i(t)] \quad (2)$$

where $0 < \alpha(t) < 1$ is the learning-rate function and $h_{j,i}(t)$ is the neighborhood function often taken as a Gaussian function. Both $\alpha(t)$ and the width of $h_{j,i}(t)$ decrease gradually with the increasing step t.

Because of the way it computes the Euclidean distance between the input pattern and map units, the traditional SOM training algorithm can handle only numeric values and is unable to directly process categorical values. In case that a categorical attribute is involved in a dataset, a popular approach is to perform attribute transformation. In Section II-B, we describe the transformation process and discuss the shortcomings of this approach.

B. Problem With the Conventional Approach

Typical approaches by which the conventional SOM handles categorical attributes are some sorts of binary encodings that transform each categorical attribute to a set of binary, and thus numeric, attributes before training an SOM starts. Specifically, a popular technique is to transform a categorical attribute to binary attributes according to its domain such that each distinct categorical value becomes a binary attribute in the new dataset. For an illustrative example as shown in Fig. 1, suppose the domain of the attribute *Favorite_Drink* is { Coke, Pepsi, Mocca, Nescafe }, then *Favorite_Drink* is transformed to a set of attributes *Coke*, *Pepsi*, *Mocca*, and *Nescafe*. As indicated in Fig. 1, a data pattern having Coke as the value of *Favorite_Drink* has 1 marked in the *Coke* attribute of the transformed relation, and the other three attributes (*Pepsi*, *Mocca*, and *Nescafe*) are set to 0.

This approach has the following four drawbacks: 1) It is unable to determine the similarity information among categorical values. For example, the transformed relation does not show that Coke is more similar to Pepsi than Mocca. 2) When the domain of a categorical attribute is large, transforming it to a set of binary attributes increases the dimensionality of the relation, resulting in wasting storage space and increasing training

Name	Favorite_Drink	Amount
Jane	Coke	7
Mary	Pepsi	7
Tom	Mocca	7
Helen	Nescafe	7

$\Rightarrow$

Name	Coke	Pepsi	Mocca	Nescafe	Amount
Jane	1	0	0	0	7
Mary	0	1	0	0	7
Tom	0	0	1	0	7
Helen	0	0	0	1	7

Fig. 1. Categorical attribute *Favorite_Drink* is transformed to four binary attributes according to its domain.

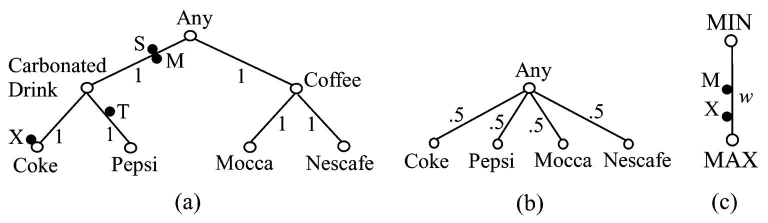

Fig. 2. (a) Distance hierarchy with link weight 1. (b) Two-level distance hierarchy for simple matching approach. (c) Degenerated distance hierarchy with link weight $w = (\max_A - \min_A)$ for a numeric attribute A.

time. 3) It is hard to maintain the new schema. When the domain of an attribute changes, the transformed relation schema needs to be changed. For instance, if "Juice" is added to the domain of the *Favorite_Drink*, an additional attribute *Juice* needs to be included in the transformed relation schema. 4) New binary attributes are unable to reflect the semantics of the original attribute. For example, after the transformation, the four binary attributes do not express the meaning of *Favorite_Drink*.

III. Generalizing the Self-Organizing Map

We first introduce distance hierarchy, which enables direct expression of the distance between categorical values. Then, a generalized self-organizing map model is proposed, and the approach to training a GSOM is presented.

A. Distance Hierarchy

To overcome the drawbacks of the conventional approach, we propose distance hierarchy, a concept hierarchy [30] and [31] extended with link weights as shown in Fig. 2. The distance hierarchy facilitates the representation and computation of the distance between categorical values. In fact, it is a general distance representation mechanism for both categorical and numeric values. This argument will be discussed later in Section III-D.

A concept hierarchy is composed of concept nodes and links, where higher-level nodes represent more general concepts while lower-level nodes represent more specific concepts. We extend the structure with link weights: Each link has a weight representing a distance. The distance between two concepts at leaf nodes is then defined as the total link weight between those two leaf nodes.

Link weights are assigned by domain experts. There are several assignment alternatives. The simplest way is to assign all links a constant weight, e.g., 1. Other alternatives include considering different weight assignments according to the levels where nodes reside. For instance, an option is to assign heavier weights to the links closer to the root and lighter weights to the links further away from the root. For simplicity, unless stated explicitly, each link weight is set to 1 in this paper.

A point X in a distance hierarchy consists of two parts, an *anchor* and a positive real-valued *offset*, denoted as (N_X, d_X), where the anchor is a leaf node and the offset represents the distance from the root of the hierarchy to X. Functions *anchor* and *offset* return these two elements, respectively; that is, *anchor* $(X) = N_X$ and *offset* $(X) = d_X$. For example, in Fig. 2(a), assume $M = (\text{Pepsi}, 0.3)$ indicating that M is in the path of (Pepsi, Any) and is 0.3 away from the root Any. Moreover, *anchor* $(M) =$ Pepsi, and *offset* $(M) = 0.3$.

Definition (Equivalent Points): Two points, X and Y, in a distance hierarchy are *equivalent*, denoted as $X \equiv Y$, if they are at the same position in the hierarchy; that is, given $X = (N_X, d_X)$ and $Y = (N_Y, d_Y)$, $X \equiv Y$ if $N_X = N_Y$ and $d_X = d_Y$.

A point X is an *ancestor* of a point Y if X is in the path from Y to the root. A *least common ancestor* of two points X and Y, denoted as $LCA(X, Y)$, is defined as a point at the deepest tree node that is an ancestor of X and Y. Since all the points at that tree node are equivalent, we will use the tree node label as the least common ancestor directly for simplicity if there is no confusion. The *least common point* of X and Y, denoted as $LCP(X, Y)$, is defined as one of the three cases: 1) either X or Y if $X \equiv Y$; 2) Y if Y is an ancestor of X; otherwise, 3) $LCA(X, Y)$.

For example, in Fig. 2(a), points M and S are equivalent, and they are ancestors of points X and T. $LCA(X, T)$ is a point at the node Carbonated_Drink (or Carbonated_Drink for simplicity). $LCP(M, S)$ is either M or S since they are equivalent. $LCP(M, X) = M$ since M is an ancestor of X. $LCP(X, T)$ is Carbonated_Drink since X and T are not equivalent, and none of them are the ancestors of the others

Two points at the same position are equivalent. However, equivalent points in a distance hierarchy unnecessarily have identical values. In particular, they may have distinct anchors, as shown in Theorem 1. For example, in the case of $S = (\text{Coke}, 0.3)$ and $M = (\text{Pepsi}, 0.3)$, S is equivalent to M.

Theorem 1: Given $X = (N_X, d_X)$, $Y = (N_Y, d_Y)$, and $N_X \neq N_Y$, if $d_{LCA(N_X, N_Y)} \geq d_x$ and $d_X = d_Y$, then X and Y are equivalent, i.e., $X \equiv Y$.

Proof: Let $LCA(N_X, N_Y) = A$. Since $d_{LCA(N_X,N_Y)} = d_A \geq d_x$, X is in the path between A and the root. Similarly, Y is in the same path between A and the root, too. Furthermore, because of $d_X = d_Y$, the distance from the root to X and to Y is the same; therefore, X and Y must be at the same position. In other words, $X \equiv Y$. ■

The distance between two points in a distance hierarchy is the total weight between them. Let $X = (N_X, d_X)$ and $Y = (N_Y, d_Y)$ be two points, then, the distance between X and Y is defined by

$$|X - Y| = d_X + d_Y - 2d_{LCP(X,Y)} \tag{3}$$

where $d_{LCP(X,Y)}$ is the distance between the root and the least common point of X and Y. According to (3) and the definition of equivalent points, the distance between equivalent points is obviously equal to zero.

For example, in Fig. 2(a), assume that $X = (\text{Coke}, 2)$, $M = (\text{Pepsi}, 0.3)$, $S = (\text{Coke}, 0.3)$ and $T = (\text{Pepsi}, 1.3)$, the distance between M and S is zero since they are equivalent. The distance between T and M is $(1.3+0.3-2*0.3) = 1$. Because $LCP(X,T)$ is the Carbonated_Drink and $d_{LCP(X,T)} = 1$, the distance between X and T is, therefore, $(2.0+1.3-2*1) = 1.3$.

A special type of distance hierarchy, called numeric distance hierarchy, for a numeric attribute, is a degenerated one which consists of only two nodes and a link, as shown in Fig. 2(c). The two nodes are the root labeled by MIN and the leaf labeled by MAX. The only link has the weight w that is equal to the range of the numeric attribute, say A; that is, $w = (\max_A - \min_A)$. A point X in a numeric distance hierarchy has the value of (MAX, d_X) where the anchor is always the MAX and the offset d_X is the distance from X to the root MIN.

B. Mapping Data to Distance Hierarchies

In this section, we discuss how to map a multidimensional data pattern to a set of distance hierarchies. Let $x = [x_1\ x_2\ \ldots\ x_n]$, either $x_i \in$ *Dom* (A_i) a categorical value or $x_i \in R$ a numeric value, and their associated distance hierarchies be $\{dh_1, dh_2, \ldots, dh_n\}$. We first describe the association between an attribute and its corresponding distance hierarchy. A categorical attribute A_i associates with the distance hierarchy dh_i in the way that the domain of A_i corresponds with the label set of the leaf nodes of dh_i, denoted as *Leaf* (dh_i), i.e., $\forall\ x_i \in$ *Dom* (A_i), $x_i \in$ *Leaf* (dh_i) and vice versa, or *Dom* $(A_i) =$ *Leaf* (dh_i). A numeric attribute A_i associates with the distance hierarchy dh_i, which is a numeric one having a root MIN, a leaf MAX, and a link weighted by the range of A_i, i.e., $w_{Ai} = (\max_{Ai} - \min_{Ai})$.

We are now at the position to state the mapping between a data pattern and its distance hierarchies. An n-dimensional data pattern associates with the set of distance hierarchies in the way that each attribute value can be mapped to a point in its distance hierarchy. Specifically, given $x = [x_1\ x_2\ \ldots\ x_n]$ and the set of distance hierarchies $\{dh_1, dh_2, \ldots, dh_n\}$, for a categorical A_i with the value x_i (i.e., $A_i = x_i$), $dh_i(x_i)$ maps the value x_i to a point X in dh_i where X has the value $(N_X, d_X) = (x_i, d_{xi})$. In other words, the mapping point X is at the leaf node labeled by the same value x_i. Recall that the domain of A_i and the leaf label set of dh_i are identical. The offset d_X, the distance from the root to X, is, therefore, the distance from the root to the leaf node x_i, i.e., $d_X = d_{xi}$.

In the case of a numeric A_i with a value x_i, the mapping $dh_i(x_i)$ maps x_i to the point X with the value $(\text{MAX}, x_i - \min_{Ai})$ in its degenerated dh_i, where MAX is the only leaf and $\min_{\text{Ai}}$ is the minimum value of A_i.

For example, suppose $x = [\text{Coke } 7]$, dh_{FD} as shown in Fig. 2(a) and dh_{Amt} as shown in Fig. 2(c) with $\max_{\text{Amt}} = 10$ and $\min_{\text{Amt}} = 0$, the mapping $dh_{\text{FD}}(\text{Coke})$ is the point X on the node Coke in dh_{FD}, i.e., $dh_{\text{FD}}(\text{Coke}) = (\text{Coke}, 2)$. Note that for brevity, FD is used for *Favorite_Drink* and Amt for *Amount*. The mapping $dh_{\text{Amt}}(7)$ is a point with the value (MAX, 7) in dh_{Amt}, i.e., $dh_{\text{Amt}}(7) = (\text{MAX}, 7)$.

C. Measuring the Distance Between Patterns

The distance between two data patterns is defined according to the mapping points in their associated distance hierarchies. All attribute values of two patterns are mapped to their hierarchies, and then, the distances between correspondent mapping points are aggregated.

Let two n-dimensional patterns $x = [x_1\ x_2\ \ldots\ x_n]$ and $y = [y_1\ y_2\ \ldots\ y_n]$ and their associated distance hierarchies be $\{dh_1, dh_2, \ldots, dh_n\}$. The distance between x and y refers to the total distance of all corresponding mapping points in the hierarchies. Formally, the distance between x and y is defined by

$$\begin{aligned} d_L(x,y) &= \left(\sum_{i=1,n} w_i (x_i - y_i)^L\right)^{1/L} \\ &= \left(\sum_{i=1,n} w_i |dh_i(x_i) - dh_i(y_i)|^L\right)^{1/L} \end{aligned} \tag{4}$$

where $dh_i(x_i)$ and $dh_i(y_i)$ are the mappings of x_i and y_i to dh_i. An attribute weight w_i allows assigning a different weight to a specific attribute. L is an integer constant. In case of $L = 1$, the metric is similar to a weighted Manhattan distance. If $L = 2$, the metric is similar to a weighted Euclidean distance.

The illustrative example of Fig. 1 is continued. We assume $x = [\text{Coke } 7]$, $y = [\text{Pepsi } 7]$, and $z = [\text{Mocca } 7]$, the distance between x and y with respect to dh_{FD} [see Fig. 2(a)] and dh_{Amt} [see Fig. 2(c)], w_i set to 1, and $L = 2$ calculated as follows. The distance of the first components of x and y after mapping is $|(\text{Coke}, 2) - (\text{Pepsi}, 2)| = 2$. The distance of the second components of x and y is $|(\text{MAX}, 7) - (\text{MAX}, 7)| = 0$. Therefore, $d_2(x,y) = (2^2 + 0^2)^{1/2} = 2$. Similarly, the distance between x and z is $d_2(x,z) = (4^2 + 0^2)^{1/2} = 4$. These three patterns are correspondent to those in Fig. 1. We can see that the distance measured via distance hierarchies is able to distinguish the different degree of the similarities among them. On the contrary, the conventional Euclidean distance via binary transformation is not. In addition, our scheme provides a unified platform for measuring the distance of the mixed-type data.

In terms of implementation, for a numeric attribute i, $dh_i(x_i) - dh_i(y_i)$ is equal to $x_i - y_i$ since $dh_i(x_i) - dh_i(y_i) = (\text{MAX}, x_i - \min_\text{i}) - (\text{MAX}, y_i - \min_\text{i}) = (\text{x}_\text{i} - \text{y}_\text{i})$.

```
Input: an $n$-dimensional training dataset D, a GSOM, and a set of $n$ distance hierarchies
Output: a trained GSOM

1. Initialize each unit m of the GSOM. (See Sec. 3.7)
2. For each pattern x in D,
    2.1 Identify its best matching unit from the GSOM. (See Sec. 3.6)
        • Map the components of x and m to their distance hierarchies.
        • Aggregate the distances of the mapping points in the hierarchies.
        • Identify m that gives the minimum distance, d, to x as the best matching unit.
    2.2 Adjust the BMU and its neighbors. (See Sec. 3.6)
        • Calculate the adjustment by multiplying d with a learning rate and a neighborhood
          function.
  Repeat till stop criterion met.
```

Fig. 3. GSOM training algorithm.

Normalization can be performed to handle the scaling problem of attributes. The MIN-MAX normalization can be adapted. The value range of an attribute, numeric or categorical, is measured in terms of the mapping points in its distance hierarchy. Specifically, the value range is the maximum distance in its hierarchy. For a numeric attribute, the range is the distance of the mapping points on MAX and MIN. For a categorical attribute, the range is the distance of the two mapping points that are most apart in the hierarchy.

D. Unifying Traditional Approaches

The proposed scheme unifies several traditional approaches of handling categorical data and numeric data, including simple matching, binary encoding, and numeric value subtraction. In other words, the scheme handles the categorical and the numeric data in a unified manner.

The simple matching approach, in which two identical categorical values result in a distance 0 or, otherwise, a distance 1, can be treated as a special case as follows. Each categorical attribute, say A, associates to a two-level distant hierarchy dh_A, which has all the domain values as the leaves of dh_A and each link weight set to 0.5 [see Fig. 2(b)]. Consequently, two patterns with an identical categorical value x_A map x_A to the same position in dh_A and result in a distance 0. Otherwise, two distinct values are mapped to two different leaves and result in a distance 1.

The binary encoding approach, in which each categorical attribute is transformed to a set of binary attributes, can be modeled by the proposed scheme in the following way: Transforming the categorical attributes and then associating each new binary attribute with a numeric distance hierarchy that has a root MIN, a leaf MAX, and a link with weight 1. The minimum and maximum values of the binary attribute are 0 and 1, respectively.

The value subtraction approach, in which subtraction operation applies directly to two numeric values, can be modeled by mapping the values to their numeric distance hierarchy [see Fig. 2(c)] and then measuring the distance between the mapping points.

E. The Generalized SOM

Given an n-dimensional dataset D, a generalized SOM model contains a map of n-dimensional units and a set of distance hierarchies $DH = \{dh_1, dh_2, \ldots, dh_n\}$. Each attribute A_i of the dataset is associated with dh_i in the manner as mentioned in Section III-B. That is, for a categorical A_i, *Dom* $(A_i) =$ *Leaf* (dh_i) and each value x_i can be mapped to a point X at the leaf labeled by x_i. For a numeric A_i, dh_i contains only the root MIN, one leaf MAX and one link with weight $w = \max_{Ai} - \min_{Ai}$. An attribute value v is then mapped to the position that is below MIN by the offset $(v - \min_{Ai})$.

For the dataset D, an associated two-dimensional generalized self-organizing map consists of a set of GSOM units arranged in a two-dimensional grid. Let m be a unit of the GSOM, the ith component m_i corresponds to the ith attribute A_i of D, which could be a numeric or a categorical attribute. We call a component of a GSOM unit either categorical or numeric, depending on what type of the attribute the component associates with. Each component is composed of two parts, $m_i = (N_i, d_i)$, where N_i is a symbolic value and d_i is a real number. For a categorical m_i, N_i shares the same domain with A_i, i.e., *Dom* $(N_i) =$ *Dom* (A_i). The numeric value d_i is in the range between 0 and the height of dh_i, i.e., $0 \leq d_i \leq$ height (dh_i). For a numeric m_i, the symbolic value N_i is always MAX and d_i has the same range as A_i, i.e., $m_i = (N_i, d_i) = (\text{MAX}, d_i)$ subject to $\min_{Ai} \leq d_i \leq \max_{Ai}$.

Each component m_i is also associated with the same distance hierarchy dh_i as A_i and can also be mapped to a point in dh_i. For a categorical m_i with the value (N_i, d_i), m_i is mapped to a point M with the same value; that is, $dh_i(m_i) = M = (N_i, d_i)$. Recall that the symbolic part of a categorical component shares the same domain with A_i and, therefore, the same with the leaf set of dh_i. Similarly, for a numeric m_i with the value (MAX, d_i), m_i is mapped to a point M in dh_i such that its anchor is the leaf node MAX and M is below MIN by $d_i - \min_{Ai}$, i.e., $M = (\text{MAX}, d_i - \min_{Ai})$.

We continue the example in Section III-B. Let $m = [(\text{Pepsi}, 0.3)\ (\text{MAX}, 5)]$ be a unit of the GSOM with the components correspondent to the attributes of the input data. Similarly, we can map each component of m to its associated hierarchy. m_1 is mapped to a point M at the position of (Pepsi, 0.3) in dh_{FD}, and m_2 is mapped to a point at the position of (MAX, 5) in dh_{Amt}.

F. Training a GSOM

Given an n-dimensional dataset D, a GSOM and a set of n distance hierarchies, the GSOM training algorithm is outlined in Fig. 3.

Like training a conventional SOM, two key steps are identified in training a GSOM: 1) determining the BMU, and 2) updating the BMU and its neighborhood units. The BMU is selected from the units such that it has the minimum distance to the input training pattern. The distance between a training pattern x and a GSOM unit m is computed as the root of the total squared difference between the correspondent components of x and m. The difference between the components of x and m is computed by mapping the components to their hierarchies and then measuring the distance in the hierarchy. Formally, $d(x,m) = (\sum[dh_i(x_i) - dh_i(m_i)]^2)^{1/2}$ where $dh_i(x_i)$ and $dh_i(m_i)$ map x_i and m_i to dh_i, respectively.

We continue the example in Section III-E. The distance between $x =$ [Coke 7] and $m =$ [(Pepsi, 0.3) (MAX, 5)] is calculated as follows. The distance of the first components of x and m after mapping is $|(\text{Coke, } 2) - (\text{Pepsi, } 0.3)| = 1.7$. The distance of the second components of x and m is $|(\text{MAX, } 7) - (\text{MAX, } 5)| = 2$. The distance between x and m is, therefore, $[(1.7)^2 + (2)^2)]^{1/2} = 2.62$.

Once the distance between a training pattern and the BMU is determined, the adjustment amounts of the BMU and its neighbors are computed by multiplying the distance by a learning rate and a neighborhood function. Then, the BMU and its neighbors are adjusted by their respective adjustment amount such that the adjusted units become closer toward the training pattern. The adjusting of each component m_i of a GSOM unit m toward its correspondent attribute x_i of a training pattern x is accomplished by moving the mapping point M of m_i in dh_i toward the mapping point X of x_i.

Note that in the context of training a GSOM, it is always a point adjusted toward the other point that is located at a leaf node, because the value of a categorical attribute of a training pattern is always mapped to a leaf node. Therefore, for simplicity, we limit our discussion of adjusting a point to this situation.

During the adjusting phase, x_i is the adjusting aim of m_i. In terms of the points in their associated hierarchy, the mapping points, say X and M, of x_i and m_i form the adjustment path where M moves toward X along the path during adjusting. In particular, referring to Fig. 2(a), let X be the mapping point which M and T move toward, the anchors of X, M, and T be C (Coke), P (Pepsi), and P (Pepsi), respectively, the adjustment amount be δ, and N_{LCA} be the least common ancestor of C and P. In this example, N_{LCA} of C and P is the Carbonated_Drink. Case 1: If M is an ancestor of N_{LCA} and it does not cross N_{LCA} after adjustment, then the new M is $(P, d_M + \delta)$, where d_M is the offset of M to the root. Case 2: If M is an ancestor of N_{LCA} and it crosses N_{LCA} after adjustment, then the new M is $(C,\ d_M + \delta)$. Case 3: If N_{LCA} is an ancestor of T and T does not cross N_{LCA} after adjustment, then the new T is $(P, d_T - \delta)$. Case 4: If N_{LCA} is an ancestor of T and T crosses N_{LCA} after adjustment, then the new T is $(C, 2d_{N_{LCA}} - d_T + \delta)$. Note that whenever the adjusted point crosses its least common ancestor N_{LCA}, the point changes its anchor to the anchor of the other point that it moves toward. For instance, new M and new T in Case 2 and 4 have changed their anchors.

Followed is an illustration of the adjusting process. Assume that x_i is mapped to the point $X = (\text{Coke, } 2)$ in Fig. 2(a). After computation, the adjustment amount to a unit component m_i is assumed to be $\delta = 0.9$. If m_i is mapped to the point $M = (\text{Pepsi, } 0.3)$, then M moves toward X and becomes (Coke, 1.2) after adjustment. Accordingly, after the adjustment, new m_i becomes (Coke, 1.2). If m_i is mapped to $T = (\text{Pepsi, } 1.3)$, then T moves upward, crosses the node Carbonated_Drink, and becomes $T = (\text{Coke, } 1.6)$. New m_i therefore becomes (Coke, 1.6). Note that both cases crossed the least comment ancestor of Coke and Pepsi (i.e., Carbonated_Drink) and hence m_i changed its anchor. If $\delta = 0.2$ instead, new M and new T do not cross the least comment ancestor. Therefore, new M and new T retain their anchors and become (Pepsi, 0.5) and (Pepsi, 1.1), respectively.

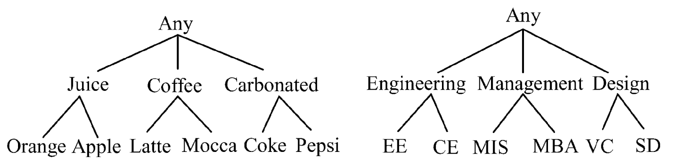

Fig. 4. Distance hierarchies for the synthetic datasets.

TABLE I
SYNTHETIC CATEGORICAL DATASET

Group	Department	Drink	Count
1	EE	Coke	20
2	CE	Pepsi	10
3	MIS	Latte	20
4	MBA	Mocca	10
5	VC	Orange	20
6	SD	Apple	10

Adjusting a numeric m_i can be made easier because the hierarchy that m_i associates with is degenerated. Assume that $m_i = (\text{MAX, } d)$ and the adjustment amount is δ. The mapping point M of m_i should be at the position of $(\text{MAX, } d - \min_{Ai})$ in dh_i. After adjustment, M moves to the position of $(\text{MAX, } d + \delta - \min_{Ai})$. In terms of the component, m_i becomes $(\text{MAX, } d + \delta)$ after the adjustment.

G. Initializing a GSOM

To initialize a GSOM, each unit is set to a random vector in its valid range. In terms of the mapping points in their distance hierarchies, each component of a GSOM unit is initially associated to a random position in its hierarchy. In particular, for a categorical $m_i = (N_i, d_i)$ associated with the attribute A_i of the training data, the symbolic part N_i is set to a value randomly chosen from the domain of A_i, i.e., $N_i = x_i \in Dom(A_i)$, which is associated with the node labeled x_i in dh_i. The numeric part d_i shall be randomly set in the range between 0 and the value from the root to the node x_i, i.e., $0 \leq d_i \leq d_{xi}$. For a numeric $m_i = (N_i, d_i)$ associated with A_i, N_i is set to the value MAX and d_i to a random number between the minimum and the maximum of A_i, i.e., $\min_{Ai} \leq d_i \leq \max_{Ai}$. Therefore, when mapped, m_i will be positioned at a location between its anchor and the root of dh_i.

IV. EXPERIMENTS

We conducted experiments on several synthetic datasets, and the results from different datasets showed consistent outcomes. We present the results of two synthetic datasets, depicting the

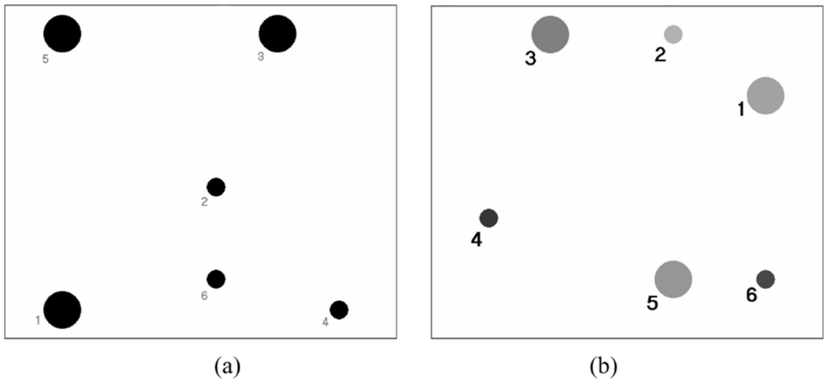

Fig. 5. Training results of a 100-unit GSOM using (a) binary encoding for categorical attributes and (b) categorical attributes directly.

effectiveness of the GSOM and the comparison with the conventional SOM. We also apply the GSOM to analyze a real dataset.

Our prototype GSOM system was developed using Borland C++ Builder. The map is two dimensional, and its units are arranged on a rectangular lattice. The distance hierarchies for categorical attributes used in the experiments were constructed manually, and the weight of each link of the distance hierarchies was set to one. Fig. 4 shows the categorical distance hierarchies for the synthetic datasets.

A. Synthetic Categorical Data

This experiment aims to show the effectiveness of the GSOM for handling categorical data and to compare its performance to that of the conventional SOM. The experiment consists of three parts. The first part used a GSOM with the categorical attributes being binary-encoded. The second part used a GSOM with the categorical attributes being used directly. The third part used the conventional SOM toolbox [32] with the categorical attributes being binary-encoded. The toolbox was developed by the Laboratory of Information and Computer Science, Helsinki University of Technology, Helsinki.

Table I shows the first synthetic dataset, which has six groups of two categorical attributes including *Department* and *Drink*. The total number of data patterns is 90, and each group has 10 or 20 patterns. The patterns in Group 1 and 2 are students from the Engineering College and all like carbonated drinks. The patterns in Group 3 and 4 are students from the Management College who like coffee. The patterns in Group 5 and 6 are students from the Design College who like juice. Obviously, if we take into consideration the similarity embedded between categorical values, the patterns in Group 1 and 2 are then more similar to each other than to the patterns in other groups. Similar situations occur in the patterns in Group 3 and 4, and the patterns in Group 5 and 6.

The first part of the experiment uses a GSOM with the categorical attributes being transformed to a set of binary attributes. The map size is 100 units. The learning rate is set to a linear function $\alpha(t) = \alpha(0) * (1.0 - t/T)$ with the initial value $\alpha(0) = 0.9$ and the training time T at least 10 times of the map size. The neighborhood function is set to a Gaussian function.

The training results are shown in Fig. 5(a). The spots indicate where the training patterns are assigned. The spot size is proportional to the number of patterns assigned to the spot. Clearly, the patterns are assigned to six groups on the trained GSOM. Inspecting the patterns assigned in the units, we find that each of the groups in the data space naturally corresponds to a group on the map. That means the patterns from the same data group are assigned to the same unit of the map. However, this trained map does not reflect the topological order that we embedded among the groups in the dataset. For instance, Group 5 is located at the upper-left part while Group 6 is located at the lower-middle of the trained map. Although their patterns are close in the data space, they are not next to each other on the map.

The second part of the experiment uses the GSOM with the categorical values being used directly. The training parameters are set to the same as those of the first part. Fig. 5(b) shows the results indicating that the training patterns forms six groups on the map. Inspecting further, we found that the patterns from a group in the data space is also correctly assigned to the same unit. In contrast to Fig. 5(a), similar groups in the dataset come next to each other on the map. For example, in Group 1 and 2, both the students of the Engineering College with carbonated drinks, come close to each other on the upper-right of Fig. 5(b). Similarly, Group 3 and 4, as well as Group 5 and 6, are next to each other on the map. This phenomenon does not appear on the maps of the first and the third parts of the experiment, in which both transform the categorical values to binary values before training.

The third part of the experiment uses the SOM toolbox developed in the Helsinki University of Technology, referred to as KSOM in this paper. We run the KSOM with the same size of the previous experiment (i.e., 100 units), and the categorical attributes are transformed to binary attributes. The training results are shown in Fig. 6. The six white spots on the trained KSOM indicate that the input patterns are clustered into six groups. Examining the details, we found that each group on the trained KSOM also corresponds to a group in the dataset. However, like the results of the first part, no special topological order is found among the six groups. For instance, Group 1 is located at the lower-left part of the map while Group 2 is far away at the upper-middle.

When trained using categorical attributes directly, our GSOM can show the categorical value distribution of the prototype vectors associated with individual GSOM units as in Fig. 7. From this kind of map, we can tell how the categorical values spread out on the map. We use the color of a unit to show the symbolic

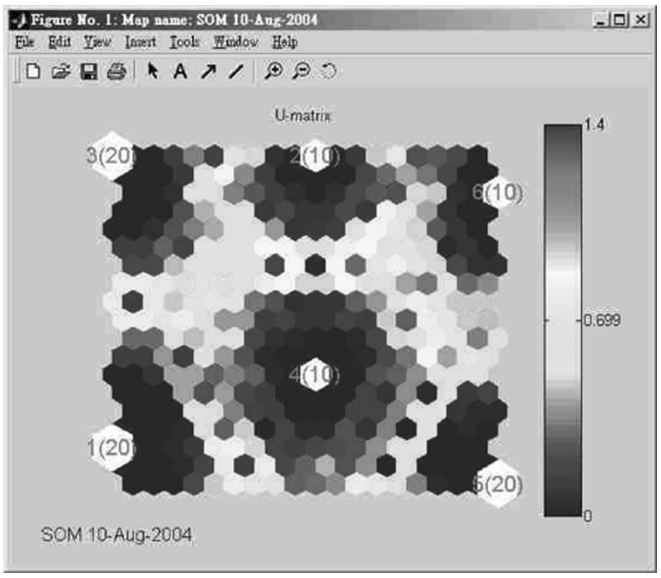
Figure No. 1: Map name: SOM 10-Aug-2004
File Edit View Insert Tools Window Help
U-matrix
3(20)
2(10)
6(10)
4(10)
1(20)
5(20)
1.4
0.699
0
SOM 10-Aug-2004

Fig. 6. Training results of a 100-unit Kohonen SOM.

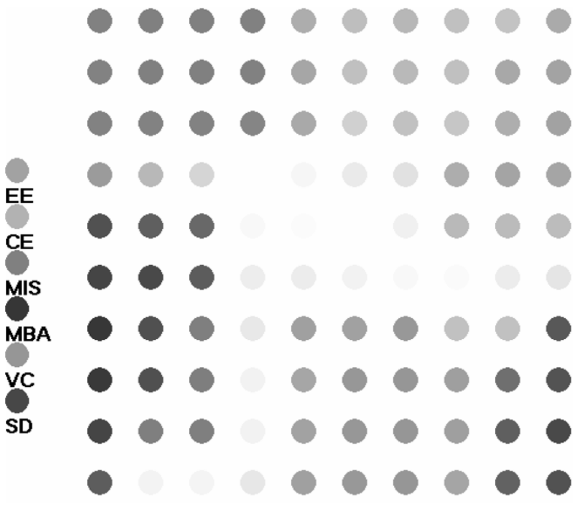

Fig. 7. Value distribution of the categorical attribute *Department* on the trained GSOM.

part N_i of a GSOM unit's component $m_i = (N_i, d_i)$ and the lightness of the color to show the numeric part d_i, which indicates how far the mapping point is in its hierarchy away from the root. The lighter the color is, the smaller the numeric value of the component is. Equivalently, lighter color represents that the mapping point is closer to the root.

Fig. 7 shows the value distribution of the attribute *Department*. As shown, the MIS and MBA of the Management College are distributed at the upper-left and the lower-left area, respectively, while the EE and CE of the Engineering College are distributed at the upper-right of the map. Note that some units, like those at (4, 3), (4, 4), and (4, 5) of the map, have lighter colors, which means that the mapping points of the units in their distance hierarchy have smaller offsets and are closer to their root. These units are usually found at the boundary where different symbolic parts, representing different groups, meet.

In various experiments, we tried different initialization of the GSOM and tried different input order of the training data to check whether the GSOM produces consistent outcomes. The experimental results showed that neither different initialization nor different input order affect the results of topological ordering.

TABLE II
SYNTHETIC MIXED DATASET

Group	Dept.	Drink	Amount (μ,σ)	Count
1	MIS	Coke	N(500, 25)	60
2	MBA	Pepsi	N(400, 20)	30
3	MBA	Pepsi	N(300, 15)	30
4	EE	Latte	N(500, 25)	60
5	CE	Mocca	N(400, 20)	30
6	CE	Mocca	N(300, 15)	30
7	SD	Apple	N(500, 25)	60
8	VC	Orange	N(400, 20)	30
9	VC	Orange	N(300, 15)	30

B. Synthetic Mixed Data

This experiment uses a synthetic mixed dataset, consisting of categorical and numeric attributes, to check the applicability of the GSOM. The synthetic dataset has 360 patterns including two categorical attributes, *Department* and *Drink*, and one numeric attribute, *Amount*. The distribution of the dataset is shown in Table II. For the *Amount* of each group, we decide an average amount μ, set the standard deviation σ to five percent of μ, and then generate an amount for each pattern in the group according to a normal distribution $N(\mu, \sigma)$. The map size is 100 units, and other parameter settings are the same with those of the first experiment.

The training results are shown in Fig. 8. Due to the inclusion of the numeric attribute, the patterns in the same group, which have identical values on *Department* and *Drink*, in the training dataset may not necessarily be assigned to the same unit on the trained GSOM. Roughly, the groups which have larger variations of the numeric values are likely to spread wider on the trained map. For example, Group 1, 4, and 7 spread wider than the others.

Fig. 8(b) shows the results of training a GSOM using the categorical attributes directly. Group 1–3 are next to one another on the map. Inspecting the patterns in the units, we found that they are all from the Management College and all like carbonated drinks. Similarly, Group 4–6, as well as Group 7–9, are gathered nearby. On the contrary, the results of training a GSOM using binary encoding for the categorical attributes are shown in Fig. 8(a), which do not reflect the correct topological order. For example, Group 7 is at the lower-right part while Group 8 and 9 are far apart at the upper-right. Moreover, Group 2 and 3 come in between them.

Note that Fig. 8(a) indicates that the numeric attribute *Amount* has significant impact on the training. The patterns with larger amounts, i.e., Group 1, 4, and 7, tend to be under the diagonal of the map while the other groups, with smaller amounts, tend to gather above the diagonal. Fig. 8(b) does not show this trait because categorical attributes in the *direct* GSOM approach assume more diverse impact during the training due to different extent of the similarity between categorical values. For instance, using the distance hierarchy in Fig. 4, the distance between Coke and Mocca is as twice as that of Coke and Pepsi. Consequently,

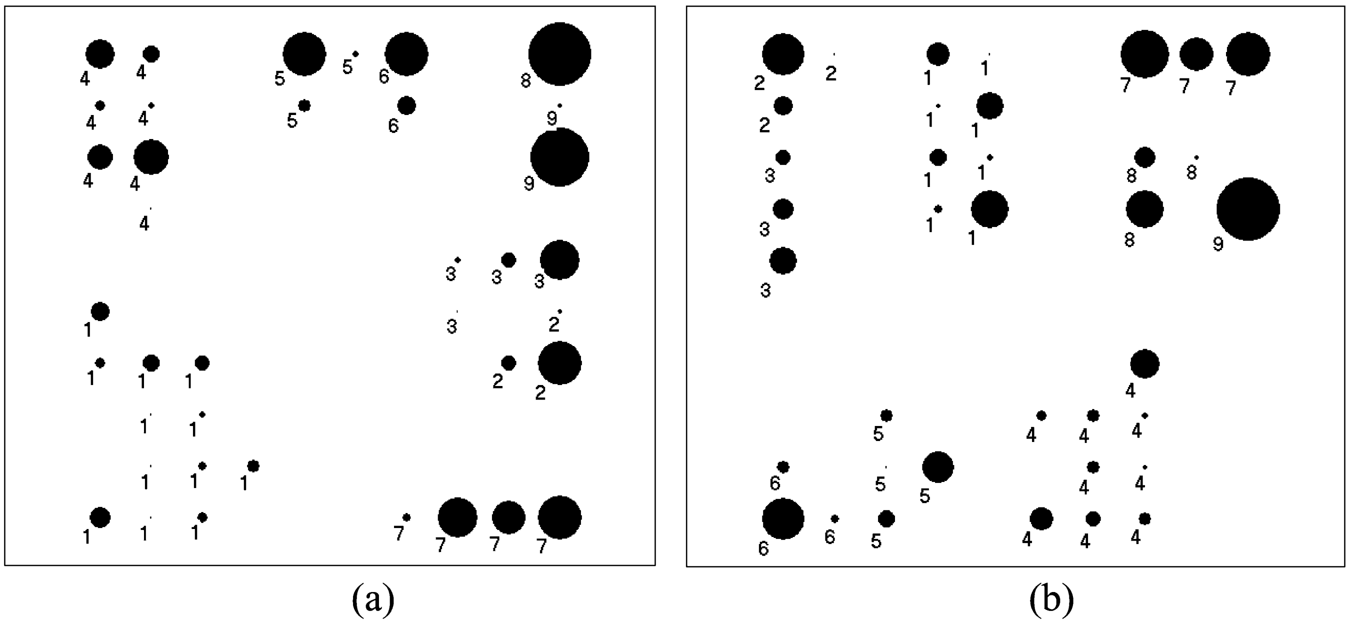

Fig. 8. Training results of a 100-unit GSOM using (a) binary encoding for categorical attributes and (b) categorical attributes directly.

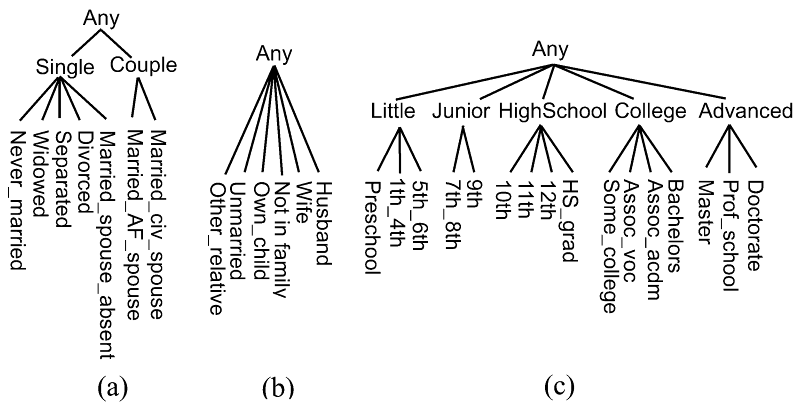

Fig. 9. Distance hierarchies for (a) *Marital_status*, (b) *Relationship*, and (c) *Education* attributes of the UCI Adult dataset.

distinct categorical values may contribute differently to the distance computation between a pattern and a GSOM unit. This is different from the *binary* GSOM approach in which two distinct categorical values always result in the same difference; hence, the numeric attribute plays relatively more important role in arranging the topological order.

C. Real Mixed Data

We applied the GSOM model to analyze the real dataset Adult from the UCI repository [33], which has 48 842 patterns of 15 attributes, including eight categorical attributes, six numerical attributes, and one class attribute. The class attribute indicates whether the salary is over 50 K. In the dataset, 76% of the patterns have the value of ≤ 50 K.

In this experiment, we randomly draw 10 000 patterns and use seven attributes which include three categorical attributes, *Marital_status*, *Relationship*, and *Education*; and four numeric attributes, *Capital_gain*, *Capital_loss*, *Age*, and *Hours_per_week*. The drawn patterns have 75.76% of ≤ 50 K, close to the distribution of the original dataset. Distance hierarchies for the categorical attributes are constructed as shown in Fig. 9.

TABLE III
SALARY DISTRIBUTION IN INDIVIDUAL CLUSTERS GROUPED ACCORDING TO THE TRAINED GSOM

C	>50K(%)	≤50K(%)	>50K	≤50K	Count
7	59.60	40.40	540	366	906
3	41.96	58.04	1538	2127	3665
4	14.30	85.70	218	1307	1525
1	5.24	94.76	52	940	992
6	3.86	96.14	40	995	1035
5	2.49	97.51	19	745	764
2	0.36	99.64	3	831	834
0	5.02	94.98	14	265	279
All	24.24	75.76	2424	7576	10000

After training a 400-unit GSOM using the categorical attributes directly, we manually group the results to seven clusters consisting of 9721 patterns in total, as shown in Fig. 10(b). The distribution of the salary in each group is shown in Table III, sorted decreasingly by the ratio of > 50 K. Group 7 has the largest ratio 59.6% of > 50 K. Group 1, 2, 5, and 6 all have much lower ratios than the original dataset. The remaining 279 patterns are treated as outliers, which are indicated by Group 0 in Table III.

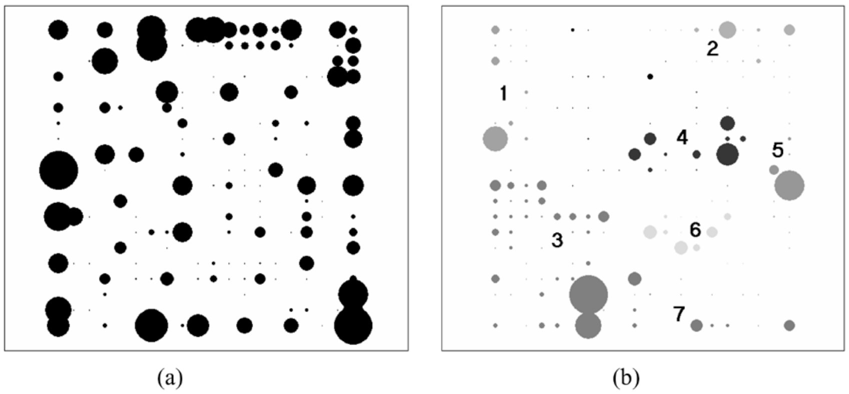

Fig. 10. Adult dataset training results of a 400-unit GSOM using (a) binary encoding for categorical attributes and (b) categorical attributes directly.

TABLE IV
MODE AND ITS PERCENTAGE OF CATEGORICAL ATTRIBUTES AND THE AVERAGE OF NUMERIC ATTRIBUTES IN INDIVIDUAL CLUSTERS ON THE TRAINED GSOM USING THE ADULT DATASET

C	Education	Marital_Status	Relationship	Age	Hours	Gain	Loss
7	Adanced (47)	Married-civ-spouse (99)	Wife (53)	42	42	3163	166
3	College (50)	Married-civ-spouse (100)	Husband (100)	44	44	1290	105
4	College (85)	Never-married (64)	Not-In-Family (99)	36	41	1116	109
1	HS (50)	Divorced (46)	Unmarried (100)	40	40	334	59
6	HS (94)	Never-married (49)	Not-In-Family (100)	39	40	271	55
5	College (98)	Never-married (93)	Own_child (100)	25	33	278	65
2	HS (92)	Never-married (89)	Own_child (86)	24	33	101	32

On the other hand, the results of using the binary encoding approach are shown in Fig. 10(a), in which pattern distribution appears relatively uniform. Consequently, it is difficult to cluster the results to seven groups.

Table IV shows the characteristics of the data patterns corresponding to the clusters on the GSOM in Fig. 10(b). Note that since the attribute *Education* has a large domain, we show the values aggregated according to Level 1 in its distance hierarchy [Fig. 9(c)]. For instance, the values Prof_school, Master, and Doctorate are generalized to the value Advanced. For a categorical attribute, the mode and its percentage in a cluster are shown. For a numeric attribute, the average value of the cluster is presented. For example, having the largest ratio of > 50 K, Cluster 7 has 47% of its patterns possessing an advanced education, 99% having the *Marital_Status* value Married-civ-spouse, and 53% having the *Relationship* value Wife. For numeric attributes, Cluster 7 has the average values *Age* 42, *Hours_per_week* 42, *Capital_gain* 3163, and *Capital_loss* 166.

Some topological order in Fig. 10(b) can be identified by inspecting Table IV. The lower-left area, where Cluster 7 and 3 reside, has the largest ratios of > 50 K and also has larger values in numeric attributes. Namely, they are older, work more hours, and have larger *Capital_gain* and *Capital_loss*. Furthermore, nearly all the patterns in that area have the *Marital_Status* value Married_civ_spouse. On the contrary, the upper-right area, where Cluster 2 and 5 reside, has the smallest ratios of > 50 K. Most of the patterns in that area are Never_married and Own_child. These patterns have smaller values in the numeric attributes. In other words, they are younger and have less work hours, smaller *Capital_gain*, and smaller *Capital_loss*. Roughly, these characteristics match our general perception regarding the household salary.

V. CONCLUDING REMARKS

Along with other experiments we conducted, the results also confirmed that the GSOM shares the following common properties with the conventional SOM: 1) When input patterns are relatively diversified, the map size should not be too small, otherwise the clustering performance will decrease. 2) The initial range of updating BMU's neighbors should cover most part of the map and then gradually decrease the range as the training time elapses. 3) Neither the initialization of the map nor the order of inputting patterns has significant impact on the topological ordering of the trained map. 4) The training time and the labeling time both linearly depends on the dataset size and the map size.

The contribution of this research is generalizing the SOM to directly handle categorical data and mixed data such that the SOM can process more diverse data and thus expand its applicability. In addition, we propose the structure distance hierarchy that can model several traditional distance computation schemes, including simple matching, binary transformation, and numeric subtraction. Furthermore, distance hierarchy offers a unified platform for measuring the distance between mixed-type, numeric, and categorical data.

ACKNOWLEDGMENT

The author would like to thank Y.-C. Lin, Y.-W. Su, and S.-H. Wang for their help in developing the prototype. The reviewers' suggestions are also appreciated.

REFERENCES

[1] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.

[2] ——, "Engineering applications of the self-organizing map," *Proc. IEEE*, vol. 84, no. 10, pp. 1358–1384, Oct. 1996.

[3] J. Vesanto, E. Alhoniemi, J. Himberg, K. Kiviluoto, and J. Parviainen, "Self-organizing map for data mining in Matlab: The SOM toolbox," *Simulation News Europe*, vol. 25, no. 54, 1999.

[4] A. Visa, "A texture classifier based on neural network principles," in *Proc. Int. Joint Conf. Neural Networks*, vol. 1, San Diego, CA, 1990, pp. 491–496.

[5] C.-H. Chang, P. Xu, R. Xiao, and T. Srikanthan, "New adaptive color quantization method based on self-organizing maps," *IEEE Trans. Neural Netw.*, vol. 16, no. 1, pp. 237–249, Jan. 2005.

[6] G. Dong and M. Xie, "Color clustering and learning for image segmentation based on neural networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 4, pp. 925–936, Jul. 2005.

[7] X. Tan, S. Chen, Z.-H. Zhou, and F. Zhang, "Recognizing partially occluded, expression variant faces from single training image per person with SOM and Soft k-NN ensemble," *IEEE Trans. Neural Netw.*, vol. 16, no. 4, pp. 875–886, Jul. 2005.

[8] M. Kasslin, J. Kangas, and O. Simula, "Process state monitoring using self-organizing maps," in *Artificial Neural Networks*, I. Aleksander and J. Taylor, Eds. Amsterdam, The Netherlands: North-Holland, 1992, vol. 2, pp. 1532–1534.

[9] O. Simula and J. Kangas, "Process monitoring and visualization using self-organizing maps," in *Neural Networks for Chemical Engineers*, A. B. Bulsari, Ed. New York: Elsevier, 1995.

[10] J. Mantysalo, K. Torkkola, and T. Kohonen, "Mapping context dependent acoustic information into context independent form by LVQ," *Speech Commun.*, vol. 14, no. 2, pp. 119–130, 1994.

[11] H. Guterman, A. Cohen, and I. Lapidot, "Unsupervised speaker recognition based on competition between self-organizing maps," *IEEE Trans. Neural Netw.*, vol. 13, no. 4, pp. 877–887, Jul. 2002.

[12] M. Vapola, O. Simula, T. Kohonen, and P. Merilainen, "Representation and identification of fault conditions of an aesthesia system by means of the self-organizing map," in *Proc. Int. Conf. Artificial Neural Networks (ICANN'94)*, vol. 1, 1994, pp. 246–249.

[13] K. Lagus, T. Honkela, S. Kaski, and T. Kohonen, "Self-organizing maps of document collections: A new approach to interactive exploration," in *Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining*, 1996, pp. 238–243.

[14] T. Kohonen, S. Kaski, K. Lagus, J. Salojarvi, J. Honkela, V. Paatero, and A. Saarela, "Self-organization of a massive document collection," *IEEE Trans. Neural Networks.*, vol. 11, no. 3, pp. 574–585, May 2000.

[15] D. R. Chen, R. F. Chang, and Y. L. Huang, "Breast cancer diagnosis using self-organizing map for sonography," *Ultrasound Med. Biol.*, vol. 1, no. 26, pp. 405–411, 2000.

[16] A. A. Kramer, D. Lee, and R. C. Axelrod, "Use of a Kohonen neural network to characterize respiratory patients for medical intervention," in *Proc. IEEE Artificial Neural Networks in Medicine and Biology (ANNIMAB-1) Conf.*, Goteborg, Sweden, 2000, pp. 192–196.

[17] A. Hirose and T. Nagashima, "Predictive self-organizing map for vector quantization of migratory signals and its application to mobile communications," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1532–1540, Nov. 2003.

[18] G. A. Barreto and A. F. R. Araujo, "Identification and control of dynamical systems using the self-organizing map," *IEEE Trans. Neural Netw.*, vol. 15, no. 5, pp. 1244–1259, Sep. 2004.

[19] M. Milano, P. Koumoutsakos, and J. Schmidhuber, "Self-organizing nets for optimization," *IEEE Trans. Neural Netw.*, vol. 15, no. 3, pp. 758–765, May 2004.

[20] N. Kasabov, D. Deng, L. Erzegovezi, M. Fedrizzi, and A. Beber, "On-line decision making and prediction of financial and macroeconomic parameters on the case study of the european monetary union," in *Proc. IEEE ICSC Symp. Neural Computation*, 2000.

[21] G. J. Deboeck, "Modeling nonlinear market dynamics for intra-day trading," *Neural-Network-World*, vol. 1, no. 10, pp. 3–27, 2000.

[22] S. Kaski, J. Sinkkonen, and J. Peltonen, "Bankruptcy analysis with self-organizing maps in learning metrics," *IEEE Trans. Neural Netw.*, vol. 12, no. 4, pp. 936–947, Jul. 2001.

[23] H. Ritter and T. Kohonen, "Self-organizing semantic maps," *Biol. Cybern.*, vol. 61, pp. 241–254, 1989.

[24] T. Nomura and T. Miyoshi, "An adaptive rule extraction with the fuzzy self-organizing map and a comparison with other methods," in *3rd Int. Symp. Uncertainty Modeling and Analysis*, 1995, pp. 311–316.

[25] A. Blanco, M. Delgado, and M. C. Pegalajar, "Extracting rules from a fuzzy/crisp recurrent neural network using a self-organizing map," *Int. J. Intell. Syst.*, vol. 15, no. 7, pp. 595–621, 2000.

[26] Y.-P. Chen, *A Hybrid Framework Using SOM and Fuzzy Theory for Textual Classification in Data Mining*. New York: Springer-Verlag, 2003, vol. 2873, Lecture Notes in Computer Science, pp. 153–167.

[27] M. Gavrilas, V. C. Sfintes, and M. N. Filimon, "Identifying typical load profiles using neural-fuzzy models," in *Proc. IEEE Power Engineering Society Transmission Distribution Conf.*, vol. 1, 2001, pp. 421–426.

[28] M.-C. Su, C.-Y. Tew, and H.-H. Chen, "Musical symbol recognition using SOM-based fuzzy systems," in *Proc. Joint 9th IFSA World Congress and 20th NAFIPS Int. Conf.*, vol. 4, 2001, pp. 2150–2153.

[29] W. Hu, D. Xie, T. Tan, and S. Maybank, "Learning activity patterns using fuzzy self-organizing neural network," *IEEE Trans. Syst., Man Cybern.*, pt. Part B, vol. 34, no. 3, pp. 1618–1626, Jun. 2004.

[30] J. Han, Y. Cai, and N. Cercone, "Data-driven discovery of quantitative rules in relational databases," *IEEE Trans. Knowl. Data Eng.*, vol. 5, no. 1, pp. 29–40, Feb. 1993.

[31] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. San Mateo, CA: Morgan Kaufmann, 2001.

[32] T. Kohonen, J. Hynninen, J. Kangas, and J. Laaksonen. (1996) SOM_PAK: The Self-Organizing Map Program Package, Rep. A31. Helsinki Univ. Tech., Lab. Comp. Inform. Sci., Espoo, Finland. [Online]. Available: http://www.cis.hut.fi/research/som_lvq_pak.shtml

[33] C. J. Merz and P. Murphy. (1996) UCI Repository of ML Databases. [Online]. Available: http://www.cs.uci.edu/~mlearn/MLRepository.html

Chung-Chian Hsu received the M.S. and Ph.D. degrees in computer science from Northwestern University, Evanston, IL, in 1988 and 1992, respectively.

He joined the Department of Information Management at National Yunlin University of Science and Technology, Taiwan, in 1993. He was the Chairman of the Department from 2000 to 2003. He is currently the Professor at the Department. Since 2002, he has also been the Director of the Information Systems Division at the Testing Center for Technological and Vocational Education, Taiwan. His research interests include data mining, machine learning, pattern recognition, information retrieval, and decision support systems.