Self-Organizing Map and Hidden Markov Model for Data Set Generation

Tadashi Ae, Kazaumasa Kioi
Hiroshima Institute of Technology
2-1-1 Miyake, Saeki-ku, Hiroshima, 731-5193 Japan
{ae, kioi}@cc.it-hiroshima.ac.jp

Abstract
We focus on sequences of the data of which a user selects from a multimedia database. These data cannot be stereotyped because user's view for them changes by each user. Therefore, we represent the structure of the multimedia database as the vector representing both user's view and the stereotyped vector. Such a vector can be classified by SOM (Self-Organizing Map). On the other hand, we introduce a technique for data set generation. If such a set consists of sequences of data, Hidden Markov Model (HMM) will be available for practical purposes. Therefore, we introduce HMM and Vector-state Markov Model (VMM) to represent the vector of user's view, and to acquire the sequence containing the change of user's view. Lastly, we will refer to an extended technique for an interactive system using the rough set theory.
Keywords: Self Organizing Map, Hidden Markov Model, Rough Set, Data Acquisition, Interactive System.

1 Introduction

There exist a lot of objects, such as pictures, music, texts, etc., around our environment. We have various feeling or impression for these objects by looking, reading or listening. The human's feeling and impression may differ from others. Additionally, his/her feeling and impression may differ according to his/her present state (situation). These differences of human's feeling or impression would occur by the difference of his/her view.

Our view is concerned with our behaviors deeply, and is very important to understand our behaviors. For example, when we classify a database, our classifications may be different by our view for data. Additionally, our next actions also may change by our view at the present state.

The view depends on human's feeling, i.e., Kansei. Therefore, the view has the subjectivity, the ambiguity, and the situation dependability. At present, a lot of methods to manage the information containing Kansei have been proposed in the field of Kansei engineering. There is the technique using Kansei words (adjective) which will be analyzed quantitatively. In this thesis, we would represent our view for an object as a vector which is gotten from evaluation values of several Kansei words (adjective).

We apply the view acquired in this way to a sequence generation. We have a view for an object, and decide a next action (data selection, etc.) with our view. Such a series of

actions constructs a sequence. There are a lot of objects which can be regarded as sequences, such as music composition, text generation, etc... Generally, it is thought that there exists the regularity in our actions, and the research for recognition of such actions (sequences) has been done actively[1]. However, these methods are not applicable to acquire the changes of our actions with our view, because human's view is not included.

We focus on the sequence which has a vector among various sequences depending on applications, and acquire the sequences of a user's data selection from the database of which elements are represented as vectors.

A research of classification and generation of sequences having a vector as an element has been done by Usui et al.[2]. They first classify the database having a stereotyped vector as an element by LVQ (Learning Vector Quantization)[3].. Next, they generate class sequences by GA (Genetic Algorithm), and evaluate the sequences according to entropy. Their method has been applied to a tour planning system (generation of city sequences).

However, when the database is a multimedia database containing pictures, music, movie, etc., in addition to the stereotyped vector, these data cannot be stereotyped because user's view for these data would change variously. Additionally, user's next action changes by the transition of his/her view. Therefore, the above method cannot be applied to sequence generation from the multimedia database. An impression for data like pictures, music and texts depends on user's view.

There is a certain order relation (user's intention) in the sequence of the multimedia data, since a user has determined the next action (data selection) based on his/her view. Therefore, it is difficult for the above method to acquire user's intention depending on his/her view.

Consequently, we represent the structure of the multimedia database as the stereotyped vector and the vector representing user's view. We propose a method of acquiring (or generating) the sequences having the structure as an element.

At present, various methods to acquire (or recognize) sequences have been proposed. The finite-state automaton is a typical model to acquire sequences. We also acquire sequences by using the automaton. However, the deterministic model requires a large number of states, because the states must be represented apparently. We consider to represent vectors representing user's view as the states, because user's view changes by his/her present state. However, user's view cannot be represented as a state apparently, because his/her view is complex and vague (We represent user's view as a vector). On the other hand, the probabilistic model does not require such a large number of states, because the state is understood to be already reduced to a state representing a nearly equivalent state. Therefore, we consider a probabilistic model.

Hidden Markov Model (HMM) is an example of the probabilistic models, and provides an efficient statistical method representing an indefinite time series data. HMM is a model which acquires the transitions between the states representing implicitly and the outputs of the symbols by the probability. In this thesis, we apply HMM to acquire symbol sequences, but states of HMM are represented as the vector representing of user's view, because user's view changes with the present state as

described above. Therefore, we use HMM where a state represented explicitly corresponds to the representative vector. We call it Vector-state Markov Model (VMM)[4].

The method to acquire sequences in this research is learned by two operations. First, we classify elements of the multimedia database (the stereotyped vector and the vector representing user's view) by Self Organizing Map (SOM) which is efficient to classify vectors. Next, we acquire sequences containing the changes of user's view from sequences represented as the gotten classes by using VMM.

We demonstrate a city-sequence generation system which reflects user's intension. The system excludes various constraints required for the conventional tour planning system. We apply VMM and HMM to this system, compare the generation by VMM with it by HMM and evaluate how much this method can acquire user's intension.

We introduce the rough set theory as a rule-base technique, which plays a role of classifying the sets of data such as the sets of "Tour".

2 Self-Organizing Map (SOM)

The SOM algorithm is defined as the following;

First, if the input to the SOM is $x(t) = [\, x_{t1}, x_{t2}, ..., x_{tn} \,]^T \in R$, where t is the discrete-time coordinate, each neuron i in the map contains a reference vector $m_i(t) = [\, m_{i1}, m_{i2}, ..., m_{in} \,]^T \in R$, which has the same number of elements as the input vector $x(t)$.

1. The initial configuration(the dimension, the number of neurons, etc.)
2. The following operation is done from t = 1 to t = T.
3. An input vector $x(t)$ is compared with all the reference vectors $m_i(t)$, and dis_i is calculated by the Euclidean distance between $x(t)$ and $m_i(t)$.

$$dis_i = \| x(t) - m_i(t) \| \tag{1}$$

4. The best-matching neuron c on the map, i.e., the neuron where the reference vector is most similar to the input vector in a metric is selected by the following equation. This best matching neuron is often called the winner.

$$c = \arg\min_i | dis_i | \tag{2}$$

5. Each reference vector $m_i(t)$ is learned by the following equation

$$\begin{aligned} m_i(t+1) &= m_i(t) + \alpha(t) h_{ci}(t)[\, x(t) - m_i(t) \,] && \text{for each } i \in \alpha(t) \\ m_i(t+1) &= m_i(t) && \text{otherwise,} \end{aligned} \tag{3}$$

where t is the discrete-time index of the variables, the factor $\alpha(t) \in [0,1]$ is a scalar that defines the relative size of the learning step, and $h_{ci}(t)$ is the following neighborhood function reduced in association with t.

$$h_{ci}(t) = \exp\left(-\frac{\|r_c - r_i\|}{2\sigma^2(t)}\right) \tag{4}$$

$r_c \in R$ is the position vector of the neuron c (winner) and $r_i \in R$ is the position vector of the neighborhood neuron i. $\sigma(t)$ is a size of the neighborhood reduced in association with t. $\alpha(t)$ and $\sigma(t)$ are defined as follows;

$$\alpha(t) = \alpha(0)\frac{T-1}{T}, \tag{5}$$

$$\sigma(t+1) = 1 + (\sigma(t) - 1)\frac{T-1}{T} \tag{6}$$

After the learning of self-organization, the process is selected on the neuron c (winner) of an input $x(t)$ and is mapped. Accordingly, similar inputs are mapped nearly and no similar inputs are mapped far.

3 Set of Cities

The elements of the database are represented as pictures, descriptive texts and the stereotyped vectors representing the information of the city. The number of cities is 92, and those cities are located in the central Europe area. The attributes of the stereotyped vector are "peace", "economy", "industry", "agriculture", "culture & literature", "fishery", "sight" and "sports & resort". The vector representing user's view corresponds to his/her subjective evaluation which can be gotten from pictures and descriptive texts. The attributes of the vector representing user's view are represented as the followings, and users evaluate the attributes at ten levels.

- Academic (ancient monuments, the site of a battle, historical museums, etc.)
- Artistic (art museums, opera houses, etc.)
- Exciting (ski, horse riding, swimming, etc.)
- Calm (hot springs, kursaal, etc.)

When we actually apply our generation method to City Sequence Generation, there are several problems. One of them is the huge size of weight-matrix caused from the number of the cites(92 cities). We must consider to make a weight matrix easily. Therefore, we simplify the matrix by the following way. First, we partition the distributed map of cities into some block

Then regarding each block as a node, the partitioned map can be represented by the graph. By this means, we can confine the next city which can be transited from the present city.

This system has been developed by using JAVA. The system is composed of the learning mode and the generation mode. We describe these modes in the next section in detail.

3.1. Structure Mode

3.1.1 Learning Mode

First, the window as Figure 1 is displayed. The window is composed of the map representing locations of cities and the area where a user inputs sample sequences.

A user selects the tour name (the case) which corresponds to sample sequences making from this time (or inputs new tour name). When the user clicks the position of a city by the mouse, the window as Figure 2 is displayed. The window is composed of pictures, descriptive text, the stereotyped vector representing the information of the city and the radar chart in which the user inputs evaluation values.

The user looks (or reads) the pictures and the text, and inputs the vector representing his/her view by the radar chart. The user repeats this operation, and makes sample city sequences.

Inputted sample sequences are learned by VMM for each user and each case.

Table 1: Cities using Tour Planning System

city name	peace	economy	industry	agriculture	culture&literature	fishery	sight	sports&report
Amsterdam	0.7	1	0.1	0.2	0.8	0.5	1	0.5
Groningen	0.4	0.4	0.4	0.8	0.6	0.2	0.3	0.7
Den-Haag	1	0.6	0.2	0.5	0.9	0.8	0.9	0.9
Frankfurt	0.4	1	0.2	0.2	0.5	0	0.4	0.2
...			...				...	

3.1.2. Generation Mode

First, the window as Figure 3 is displayed in the generation mode. A user selects the case which wants to generate, and a city which wants to visit in the first place. The learned VMM enumerates some cities to which the user should visit next. We adopt the method where the user selects the best proper city from enumerated cities rather than the method where the system generates a tour (sequence) automatically, because the adopted method can generate better sequences for a user. In other words, the method of generation is the following algorithm.

When the user wants to know about the detailed information of enumerated cities, the user can look the window as the Figure 2 by clicking the button. The user can select the best proper city by looking (or reading) the information.

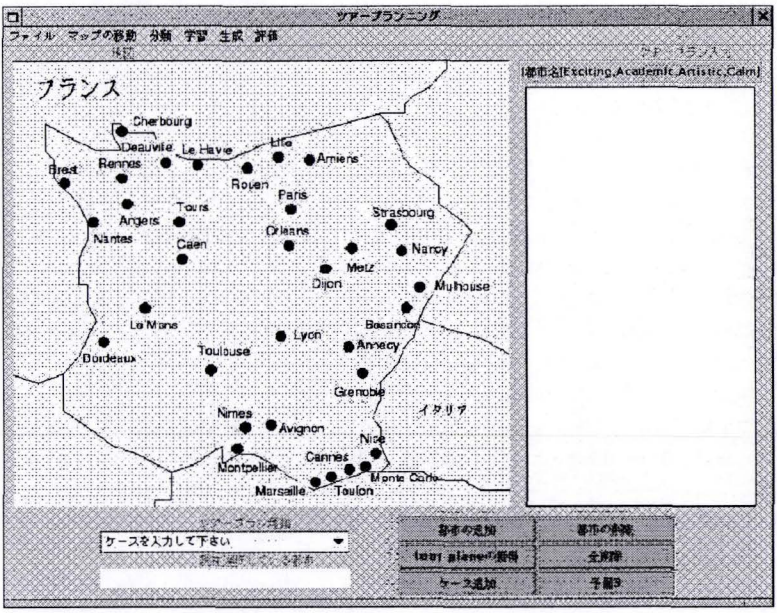

Figure 1: Window for Map

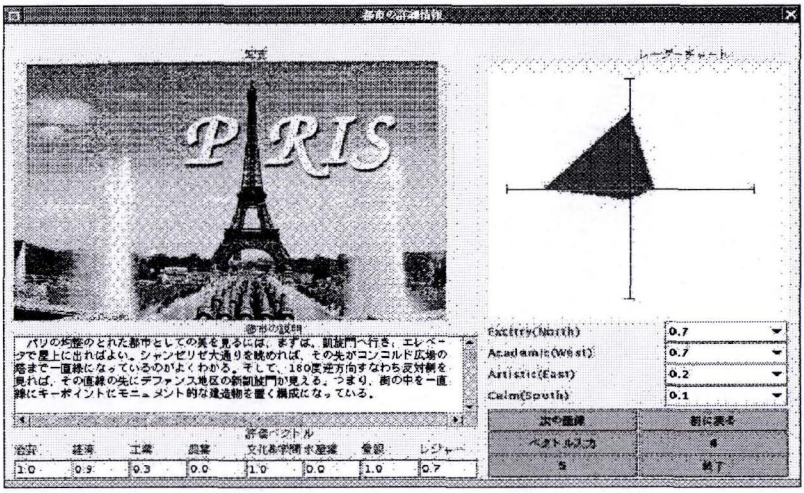

Figure 2: Window for City

Figure 3: Window for Generation

3.2. Generation of City Sequences

In this section, we show city sequences which some users generated by using above system. We compared sequences generated by HMM with them generated by VMM.

The method by HMM is used for the classes which were classified by using only the stereotyped vectors (8 dimensions), and the initial model(the number of states) and its parameters are constructed by a composer. The generation Algorithm of HMM is also used for the above method.

Three user's city sequences are shown on Tables 2 - 5. Each user generated the city sequences of two cases. We represent an example of their sequences as a sequence of the picture representing a scene of the city in Figures 4.

Generation Algorithm:

```
Generation()
{
  begin
    input of the case;
    input of the first city;
    while()
      begin
        generation of some cities which a user should visit next;
        enumeration of the cities;
        input of the best proper city;
      end
  end
}
```

Table 2: Generation by VMM (user A)

CASE	the city sequence
case1	Paris -> Bonn -> Den Haag -> Amsterdom
	Deauville -> Rothenburg -> Bremen -> Amsterdom
	Nice -> Lyon -> Marseille -> Paris
	Lyon -> Zurich -> Heidelberg -> Rothenburg
	Marseille -> Lyon -> Zurich -> Groningen
	Aachen -> Delft -> Stuttgart -> Zurich
case2	Grenoble -> Basel -> Zurich -> Stuttgar
	Bern -> Fribourg -> Heidelberg -> Bonn
	Besancon -> Lille -> Heidelberg -> Rotteldam
	Rouen -> Stuttgar -> Zurich -> Fribourg
	Avignon -> Marseille -> Besancon -> Heidelberg

Table 3: Generation by HMM (user A)

CASE	the city sequence
case1	Paris -> Groningen -> Rothenburg -> Bremen
	Deauville -> Koblenz -> Antwerpen -> Dortmund
	Nice -> Avignon -> Besancon -> Groningen
	Lyon -> Luzern -> Wiesbaden -> Stuttgart -> Marseille
	Marseille -> Dijon -> Koblenz -> Heidelberg
	Aachen -> Antwerpen -> Den-Haag -> Dusseldorf
case2	Grenoble -> Luzern -> Wiesbaden -> Den-Haag
	Bern -> Heidelberg -> Rothenburg -> Bremen
	Besancon -> Koblenz -> Amsterdom -> Brugge
	Rouen -> Bonn -> Delft -> Hamburg
	Avignon -> Lyon -> Lugano -> Heidelberg

Table 4: Generation by VMM (user B)

CASE	the city sequence
case1	Besancon -> Lyon -> Nice -> Montpellier
	Nantes -> Paris -> Deauville -> Tours
	Frankfurt -> Zurich -> St.Moritz -> Luzern
	Bruxelles -> Amsterdam -> Paris -> Rotteldam
	Grenoble -> Marseille -> Cannes -> Toulon
case2	Amsterdom -> Deauville -> Tours -> Paris
	Avignon -> Nice -> Dijon -> Le-Mans
	Amiens -> Deauville -> Orleans -> Cannes
	Fribourg -> Lausanne -> Lugano -> Munchen
	Paris -> Cannes -> Grenoble -> Bordeaux

Table 5: Generation by HMM (user B)

CASE	the city sequences
case1	Besancon -> Koln -> Den-Haag -> Eindhoven
	Nantes -> Namur -> Heidelberg -> Breme
	Frankfurt -> Heidelberg -> Antwerpen -> Stuttgart
	Bruxelles -> Amsterdom -> Heidelberg -> Rothenburg
	Grenoble -> Lugano -> Zurich -> Fribourg
case2	Amsterdom -> Koln -> Hamburg -> Den-Haag
	Avignon -> Lausanne -> Lyon -> St.Moritz
	Amiens -> Koblenz -> Delft -> Brugge
	Fribourg -> Wiesbaden -> Bonn -> Den-Haag
	Paris -> Brugge -> Essen -> Bremen

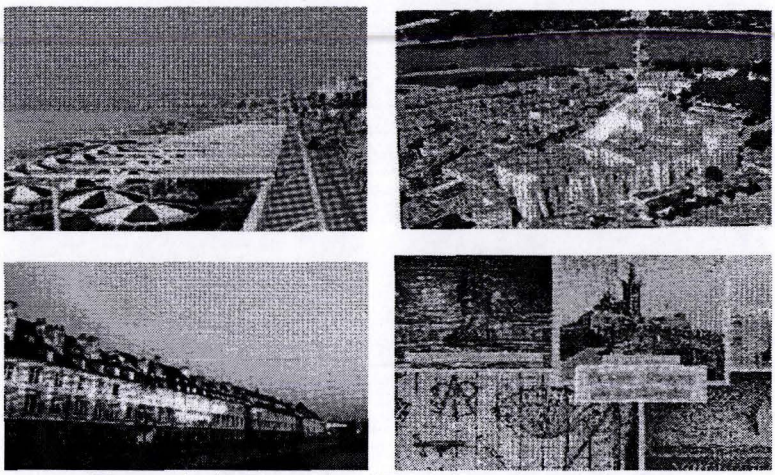

Figure 4: An Example of Tour (user A, case1)
(Nice -> Avignon -> Besancon -> Groningen)

3.3. Evaluation and Consideration

In this section, we investigate whether or not VMM can more correctly acquire a user's intention (the order relation of sequences) than HMM. Therefore, we judge from a user's evaluation for his/her sequences.

The evaluation procedure is given as follows;

1. A user generates 30 sequences (with has 4 elements) for each case by HMM.
2. A user generates 30 sequences (with has 4 elements) for each case by VMM.
3. A user looks the city informations of generated sequences and evaluates how much he/she is satisfied with the cities at ten levels.
4. A user evaluates how his/her is satisfied with the total sequences at ten levels.
5. The average of 30 sequences is represented as the evaluation values

Two user's evaluation values are shown on the Tables 6 – 9.

From these results, we find that three users evaluated the sequences generated by VMM more higher than the sequences generated by HMM at elements of the sequences and the total sequences.

Consequently, we find that VMM can acquire sequences which reflect user's intension more correctly than HMM.

Table 6: Evaluation of Generation by VMM (user A)

CASE	1 element	2 element	3 element	4 element	total
case1	8.9	7.8	8.4	7.6	7.7
case2	8.7	8.2	7.5	7.8	7.5

Table 7: Evaluation of Generation by HMM (user A)

CASE	1 element	2 element	3 element	4 element	total
case1	8.9	6.5	5.9	7.1	6.1
case2	8.7	7.1	6.2	6.9	6.4

Table 8: Evaluation of Generation by VMM (user B)

CASE	1 element	2 element	3 element	4 element	total
case1	9.1	8.2	7.7	7.6	7.7
case2	9.2	7.1	7.3	7.7	7.4

Table 9: Evaluation of Generation by HMM (user B)

CASE	1 element	2 element	3 element	4 element	total
case1	9.1	7.1	7.4	6.8	7.3
case2	8.9	7.2	6.9	6	6.7

4 Advanced Generation using Rough Set Theory

We intorduce the rough set theory as a rule-based technique, since the explicit rule representation is not usually applicable. The rough set is defined as follows[5];

For the total set U and a subset X in U, an equivalent relation R provides two sets.

$$\underline{R}X = \bigcup \left\{ Y \in U/_R : Y \subseteq X \right\} \tag{7}$$

$$\overline{R}X = \bigcup \left\{ Y \in U/_R : Y \cap X \neq \emptyset \right\} \tag{8}$$

where Equations (7) and (8) are called R-lower approximation set and R-upper approximation set, respectively.

If Equation (7) coincides with Equation (8), then X becomes an R-definable set. Otherwise, X is an R-undefinable set or an R-rough set(in short, a rough set).

In general, the set of cities is given as a rough set. The set of all cities(92 cities in our system) is U, and the generated set is X. U is classified into several equivalence sets by SOM (See Figure 5).

In general, however, the set of cities generated by the system becomes a rough set, where its meaning is given by (1) and (2).

The set generated by the system, G , is not the same as the set of user's view, V .

The difference between G and V is represented as in Figure 6, and its difference should be a minimum as much as possible. The minimization procedure is required, but we do not discuss here the details.

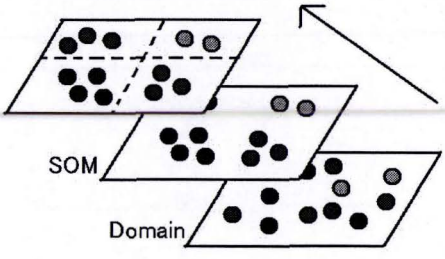

Figure5: Equivalence Sets by SOM

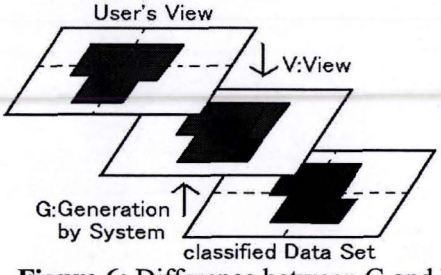

Figure 6: Difference between G and V

5 Conclusion

We discuss the classification using SOM (Self-Organization Map) and the set generation using HMM(Hidden Markov Model) or VMM (Vector-state Markov Model). As a result, VMM can show a better acquisition of user's feeling than HMM, but it is not necessarily sufficient. Therefore, we propose a method using the rough set theory, although the system is not yet completed.

References

[1] S.Aoki, M.Onishi, A.Kojima, K.Fukunaga (2001), Recognition of Behavioral Pattern Based on HMM, Trans IEICE, D-II, Vol.J85-D-II, No.7,pp.1265-1270, (in Japanese)..

[2] D.Usui, H.Araki, T.Ae (1999), Knowledge Acquisition and Generation on Structured String Data, Trans IPS of Japan, Vol.40, No.4, pp.1774-1781 (in Japanese).

[3] T.Kohonen (1989), Self-Organization and Associative Memory (3rd Ed.), springer, Verlag.

[4] T.Ae, T.Yamaguchi, E.Monden, S.Kawabata, M.Kamitani (2004), Vector representation of user's view using self-organizing map, Proceeding Electronic Imaging, SPIE vol.5298, pp.384-394.

[5] Z.Pawlak (1992), Rough Sets: Theoretical Aspects of Reasoning About Data, Kluwer Academic Pub.