Detecting Changes of Clustering Structures Using Normalized Maximum Likelihood Coding

So Hirai
Graduate School of Information Science and Technology, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo, Japan
So_Hirai@mist.i.u-tokyo.ac.jp

Kenji Yamanishi
Graduate School of Information Science and Technology, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo, Japan
yamanishi@mist.i.u-tokyo.ac.jp

ABSTRACT

We are concerned with the issue of detecting changes of clustering structures from multivariate time series. From the viewpoint of the minimum description length (MDL) principle, we propose an algorithm that tracks changes of clustering structures so that the sum of the code-length for data and that for clustering changes is minimum. Here we employ a Gaussian mixture model (GMM) as representation of clustering, and compute the code-length for data sequences using the normalized maximum likelihood (NML) coding. The proposed algorithm enables us to deal with clustering dynamics including merging, splitting, emergence, disappearance of clusters from a unifying view of the MDL principle. We empirically demonstrate using artificial data sets that our proposed method is able to detect cluster changes significantly more accurately than an existing statistical-test based method and AIC/BIC-based methods. We further use real customers' transaction data sets to demonstrate the validity of our algorithm in market analysis. We show that it is able to detect changes of customer groups, which correspond to changes of real market environments.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data mining

Keywords

Minimum Description Length principle, Normalized Maximum Likelihood, Dynamic Model Selection, Clustering

1. INTRODUCTION

1.1 Motivation and Purpose of This Paper

We are concerned with the issue of clustering multi-variate data sequences. Suppose that the nature of data changes over time. We are then specifically interested in tracking changes of clustering structures, which we call *clustering change detection* (see also evolutionary clustering [2]). This is an important issue, for example, in marketing. Suppose that each customer may be specified by her/his purchase history for a number of products, i.e., each feature shows how large each product was bought by the customer in a day. One may conduct clustering of these data to obtain a number of clusters, each of which shows a group of customers having similar purchase behaviors. The customers' data may change day by day. Tracking changes of such a time-varying clustering structure will lead to the understanding of how the organization of customer group changes over time. It is expected to give an important insight to marketing research. Figure 1.1 and 1.2 show an example of clustering change detection. Figure 1.1 shows that customers C and F change their patterns to create a new cluster. Figure 1.2 shows that new customers α and β emerge to create a new cluster.

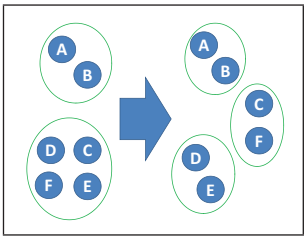

Figure 1.1: The Customers Change Their Purchase

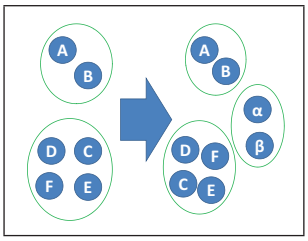

Figure 1.2: The New Customers Emerge

The purpose of this paper is twofold. One is to propose a new algorithm that tracks changes of clustering structures in the sequential setting where time series data are sequentially given and the clustering must be conducted in a sequential fashion. Although changes of clustering structures do not always imply changes of the number of clusters in general, we specifically focus on clustering changes which accompny changes of the number of clusters. We employ a Gaussian mixture model (GMM) as a representation of clustering and design the algorithm on the basis of the minimum description length (MDL) principle [10]. That is, it tracks changes of clustering structures so that the sum of the code-length for data and that for clustering changes is minimum. Here we calculate the code-length using the normalized maximum likelihood (NML) coding (see e.g.[7]). This is because the NML code-length has optimality in the sense that it achieves Shatarkov's minimax criterion [13].

The other purpose is to empirically demonstrate the va-

KDD'12, August 12–16, 2012, Beijing, China.

lidity of our proposed algorithm using both artificial and real data sets. As for the artificial data sets, we evaluate how well our algorithm works in comparison with clustering change detection algorithms using other criteria and an existing statistical-test based algorithm [14]. As for the real data sets, we employ real customers' transactions for a number of kinds of beers to conduct clustering of customers, where each cluster is a group of customers having similar purchase patterns. We investigate how well our algorithm can detect changes of customers' purchase patterns.

1.2 Previous Works

There exist a number of methods for tracking changes of clustering structures from non-stationary data sequences. Song and Wang [14] proposed a statistical-test based algorithm for dynamic clustering. It is an algorithm that estimates a GMM in an on-line manner and then conducts a statistical test to determine whether a new cluster is identical to an old one or not. If it is, the new cluster is merged into the older one, otherwise it is recognized as a cluster which has newly emerged. Sato [11] proposed an algorithm for merging and splitting of clusters in a GMM based on the variational Bayes method. Note that changes of clusters are not necessarily classified into merging or splitting. Actually some members in a number of clusters may move to generate a new cluster. Sato's algorithm cannot deal with the case.

Krempl et.al.[8] proposed a method of tracking clutering changes using the EM algorithm and Kalman filters. Our work is different from Krempl's one in that the former is concerned with changes of the number of clusters while the latter is concerned with parameter trajectories keeping the number of clusters fixed.

Yamanishi and Maruyama [16, 17] developed a theory of dynamic model selection (DMS) for tracking changes of statistical models from non-stationary data. Although this theory has been developed in a general setting for statistical model sequence selection, it can be applied to tracking of changes of clustering structures. The theory of DMS is based on the MDL principle and offers a strategy for selecting a model sequence so that the total sum of the code-length for the model sequence plus that for the data sequence relative to it is minimum. Sun et. al. [15] proposed a graph clustering algorithm called GraphScope, in which a change point for clustering was detected when the sum of code-lengths before and after it was significantly smaller than the code-length calculated without assuming that change-point. It was also designed based on the MDL principle.

1.3 Novelty and Significance of This Paper

The novelty and significance of this paper are summarized as follows:

1)*An extension of DMS into a sequential clustering setting:* In [16, 17], the theory of DMS for estimating model sequences has been explored in the batch scenario where the whole data set is given at once, and the model sequence must be detected in a retrospective way. It has remained open how to extend the DMS algorithm into the sequential scenario where data are sequentially input and the model must be selected in a sequential fashion. In this paper we extend DMS to the sequential setting and newly propose a *sequential DMS algorithm*. Every time data is input, it sequentially detects changes of clustering structures on the basis of the MDL principle so that the sum of the code-length for the data and that for the clustering change is minimum. This algorithm enables us to deal with the dynamics of clustering structures, including "merging", "splitting", "emergence", "disappearance", etc. within a unified framework from the viewpoint of the MDL principle.

2)*A new application of the RNML code-length to sequential DMS:* In the sequential DMS algorithm, the code-length for the data sequence relative to a GMM must be calculated. It is crucial how to choose a method for coding. The best choice may be the NML coding, which turns out to be the optimal code-length in the sense of minimax criterion [10]. However, it is analytically and computationally difficult to compute the NML code-length for a GMM exactly. Hirai and Yamanishi gave a method for efficiently computing an approximate variant of the NML code-length for a GMM [5]. They have recently modified it using the renormalizing technique to develop the *renormalized maximum likelihood (RNML) coding* [6]. We employ the RNML coding for the calculation of a code-length. Although it has turned out in [6] that RNML is more effective than NML in estimating the number of clusters in the batch clustering scenario, this is the first work on the application of the RNML coding to the scenario of sequential clustering change detection.

3)*Empirical demonstration of the superiority of the sequential DMS with the RNML code-length over the existing methods:* There exist a number of methods for tracking changes of clustering structures, we employ both artificial and real data sets to empirically demonstrate that our method–the sequential DMS with the RNML code-length–outperforms other methods including Song and Wang's method, AIC (Akaike's information criteria) / BIC (Bayesian information criteria)-based tracking methods etc.

4)*A novel application of clustering change detection into market research:* This paper presents a new application scenario of clustering change detection into market research. We employ a real data set consisting of customers' purchase records for a number of kinds of beers. Tracking changes of clusters of customers leads to the understanding of how customers' purchase patterns change over time and how customers move from clusters to clusters. This analysis gives a new methodology to market research.

The rest of this paper is organized as follows: Sec.2 introduces the sequential DMS algorithm. Sec.3 gives experimental results using artificial data sets. Sec.4 gives an application to market analysis. Sec.5 gives concluding remarks.

2. CLUSTERING CHANGE DETECTION

2.1 Sequential DMS Algorithm

We give a formal setting of clustering change detection. Suppose that we are sequentially given data. At each time t, we observe an n_t tuple of d-dimensional data: $X_t = (\mathbf{x}_{1t}, \ldots, \mathbf{x}_{n_t t})$ $(t = 1, \ldots, T)$, where $\mathbf{x}_{it} \in \mathbb{R}^d$ $(i = 1, \ldots, n_t)$. Let $K_t \in \mathbb{N}$ be the number of clusters at t, which we call a *model*. Let $\mathcal{Z}_t = \{1, 2, \ldots, K_t\}$. Let $Z_t = (z_{1t}, \ldots, z_{n_t t}) \in \mathcal{Z}_t^{n_t}$ $(t = 1, \ldots, T)$, where $z_{it} \in \mathcal{Z}_t$ $(i = 1, \ldots, n_t)$. We think of z_{it} as a cluster index which data $\mathbf{x}_{it}$ comes from.

For example, we may consider the case where $\mathbf{x}_{it}$ is data of the i-th customer at time t, specified by d-dimensional features (e.g., the j-th feature indicates how large the customer consumed the j-th brand beer). The cluster to which the i-th customer belongs is specified by an index z_{it}.

Dynamic model selection (DMS) is a process of estimat-

ing a model sequence $K_1 \cdots K_T$ on the basis of the MDL principle. We first give the original form of DMS that was designed for the batch scenario. Let $X^{t-1} = X_1 \cdots X_{t-1}$, $Z^{t-1} = Z_1 \cdots Z_{t-1}$ and $K^{t-1} = K_1 \cdots K_{t-1}$. Note that Z_t is supposed to be determined for X_t once K_t is given. Let $\ell(X_t, Z_t|X^{t-1}, Z^{t-1} : K_t \cdot K^{t-1})$ be the code-length for X_t and Z_t given X^{t-1},Z^{t-1}, and $K_t \cdot K^{t-1}$. Suppose that K_t is probabilistically determined by K^{t-1}. We write the code-length for K_t given K^{t-1} as $\ell(K_t|K^{t-1})$.

When a sequence X_t $(t = 1, \ldots, T)$ is given, we select an optimal sequence of models $Z^T = Z_1 \cdots Z_T$, $K^T = K_1 \cdots K_T$ so that the following criterion is minimum:

$$\sum_{t=1}^{T} \ell(X_t, Z_t|X^{t-1}, Z^{t-1} : K_t \cdot K^{t-1}) + \sum_{t=1}^{T} \ell(K_t|K^{t-1}), \quad (1)$$

where X_0, Z_0, and K_0 are initially given.

Eq.(1), which we call the DMS criterion, is the total code-length required for the encoding X^T, Z^T and K^T. Hence the process for the minimizing Eq.(1) with respect to Z^T and K^T is considered as a strategy based on the MDL principle. Note that the minimization of Eq.(1) is a batch process in the sense that Z^T and K^T must be determined in a retrospective way after the whole data X^T is observed. We are rather concerned with sequential estimation of the number of clusters K_t, i.e., every time X_t is observed, we are to choose an optimal value of K_t in a sequential fashion.

For the purpose of sequential estimation of Z_t and K_t, we approximate the minimum of Eq.(1) by locally minimizing it with respect to Z_t and K_t. That is, at each time t, letting $\hat{Z}^{t-1} = \hat{Z}_1 \cdots \hat{Z}_{t-1}$ and $\hat{K}^{t-1} = \hat{K}_1 \cdots \hat{K}_{t-1}$ be the sequence of cluster indices and the numbers of clusters obtained until $t-1$, respectively. We select an optimal model Z_t, K_t so that the following criterion is minimum:

$$\ell(X_t, Z_t|X^{t-1}, \hat{Z}^{t-1} : K_t \cdot \hat{K}^{t-1}) + \ell(K_t|\hat{K}^{t-1}). \quad (2)$$

We call this criterion the *sequential DMS criterion*. The sequential DMS algorithm is an algorithm that at each time t, takes X_t as input and outputs K_t and the resulting clustering result that minimize the criterion (2), and conducts this estimation sequentially with respect to t.

Let $\hat{Z}^T = \hat{Z}_1 \cdots \hat{Z}_T$ and $\hat{K}^T = \hat{K}_1 \cdots \hat{K}_T$ be the sequence of $\hat{Z}_t$ and $\hat{K}_t$ so that Eq.(2) is minimum at each t. Then the sum of the minimum value of Eq.(2) is given by

$$\sum_{t=1}^{T} \ell(X_t, \hat{Z}_t|X^{t-1}, \hat{Z}^{t-1} : \hat{K}_t \cdot \hat{K}^{t-1}) + \sum_{t=1}^{T} \ell(\hat{K}_t|\hat{K}^{t-1}). \quad (3)$$

This quantity is considered as an approximation of the minimum of Eq.(1) with respect to Z^T and K^T.

We employ a Gaussian mixture model (GMM) as a representation of a clustering structure. Let $f(X_t, Z_t|\theta)$ be a joint probability distribution of X_t and Z_t with parameter θ. Then the likelihood for X_t, Z_t is given as follows:

$$f(X_t, Z_t|\theta) = \prod_{j=1}^{K_t} \pi_j^{n_{tj}} \prod_{i:z_{it}=j} f(\mathbf{x}_{it}|\mu_j, \Sigma_j),$$

where we set $\theta = (\pi_j, \mu_j, \Sigma_j)$ $(j = 1, \ldots, K_t)$; π_j is the probability that $z = j$, μ_j is the mean of cluster $z = j$, and Σ_j is the variance-covariance matrix of cluster $z = j$. n_{tj} is the number of data in X_t which fall into the cluster $z = j$ $(j = 1, \ldots, K_t)$, and $f(\mathbf{x}_{it}|\mu_j, \Sigma_j)$ is a Gaussian distribution with mean μ_j and variance-covariance matrix Σ_j.

Let $\hat{\theta}(X_t, Z_t)$ be an estimator of θ from X_t and Z_t. Notice that we would like to employ the maximum likelihood estimator (MLE), but it is analytically intractable to compute it. Hence $\hat{\theta}$ may be obtained using the EM algorithm [3]. An initial value of θ in the EM algorithm is determined by $X^{t-1}, \hat{Z}^{t-1}$, and $\hat{K}^{t-1}$. We employ the renormalized maximum likelihood (RNML) code-length [6] for X_t and Z_t for the calculation of $\ell(X_t, Z_t|X^{t-1}, \hat{Z}^{t-1} : K_t \cdot \hat{K}^{t-1})$. It is defined as follows:

$$\ell_{\mathrm{RNML}}(X_t, Z_t|X^{t-1}, \hat{Z}^{t-1} : K_t \cdot \hat{K}^{t-1}; \gamma) = -\log \frac{f_{\mathrm{NML}}(X_t, Z_t; \hat{\eta}(X_t, Z_t), K_t)}{\sum_Z \int_{X \in Y'(\gamma)} f_{\mathrm{NML}}(X, Z; \hat{\eta}(X, Z), K_t) \mathrm{d}X}, \quad (4)$$

where f_{NML} is the NML distribution defined by

$$f_{\mathrm{NML}}(X_t, Z_t; \eta) = \frac{f(X_t, Z_t; \hat{\theta}(X_t, Z_t), K_t)}{\int_{X \in Y(\eta)} f(X, Z_t; \hat{\theta}(X, Z_t), K_t) \mathrm{d}X},$$

where γ is a hyper parameter, η is a parameter, and $Y(\eta)$ and $Y'(\gamma)$ denote the ranges of X for which the integrals are taken. See Sec.2.2 for details of the RNML code-length.

Below we give an algorithm for computing $\hat{\theta}(X_t, Z_t)$ and Z_t. For the sake of notational simplicity, we denote $\hat{\theta}(X_t, Z_t)$ as $\hat{\theta}^{(t)}$. Note that the number of possible changes of clustering is $\mathrm{O}(K^n)$. Hence we restrict possible changes of the number of clusters into $\{-1, 0, 1\}$. In more details, we consider the following three cases:

Case 1: $K_t = K_{t-1}$; *the number of clusters does not change*. We set $Z_t = Z_{t-1}$ and estimate $\hat{\theta}^{(t)}$ by using the EM algorithm for which the initial values of θ are set to be $\hat{\theta}^{(t-1)}$.

Case 2: $K_t = K_{t-1} - 1$; *the number of clusters decreases.* First keep the cluster indices for data in $K_{t-1} - 1$ clusters and assign the data of $z_{it-1} = K_{t-1}$ $(i = 1, \ldots, n)$ to other clusters randomly. Then run the EM algorithm with the above indices as initial values to obtain new cluster indices Z_t.

Case 3: $K_t = K_{t-1} + 1$; *the number of clusters increases.* First keep the cluster indices for data in K_{t-1} clusters and generate indices $i_1, \ldots, i_m$ $(m < n)$ randomly and set $z_{jt} = K_{t-1} + 1$ $(j = i_1, \ldots, i_m)$. Then run the EM algorithm with the above indices as initial values of Z_t to obtain a new cluster indices Z_t.

Next we consider the probability of transition from K_{t-1} to K_t. Let α be a 1-dimensional parameter. Let $K_{\max}$ be the upper bound on K_t for any t. We set the model transition probability distribution as follows:

$$P(K_1|K_0 : \alpha) = 1/K_{\max},$$
$$P(K_t|K_{t-1} : \alpha) = \begin{cases} 1-\alpha, & \text{if } K_t = K_{t-1} \text{ and } K_{t-1} \neq 1, K_{\max}, \\ 1-\alpha/2, & \text{if } K_t = K_{t-1} \text{ and } K_{t-1} = 1, K_{\max} \\ \alpha/2, & \text{if } K_t = K_{t-1} \pm 1 \end{cases} \quad (5)$$

We set K_{max} from a practical reason why an optimal K should be computed in finite memories from all possible Ks. We do not simply input K_{max} for the number of clusters because we are concerned with selecting the minimal number of clusters from among those which explain the same clustering structures. Hence we require that any clusters to which no instance contributes do not exist.

Here α should be estimated. We employ Krichevsky-Trofimov (KT) estimate [9] of α defined by

$$\hat{\alpha}_t = (N_t + 1/2)/t, \quad (6)$$

where N_t shows how many times the number of clusters has changed until time $t-1$.

For a Gaussian mixture, a single Gaussian distribution corresponds to a cluster. The increase of mixture size means that new clusters emerge and some members whose latent variables z correspond to other existing clusters move to the new ones. It includes as a special case "splitting," which means that a single Gaussian distribution is replaced with two ones. The decrease of mixture size means that an existing cluster disappears and whose z correspond to that cluster move to others. It includes as a special case "merging," which means that two Gaussian distributions are no longer discriminated and are unified into a single one. Although the change of number of clusters is restricted to $\{-1, 0, +1\}$, it would easily be extended into the case where the number of changes ranges over $\{-M, ...-1, 0, +1, ..., +M\}$ for $M > 1$.

Once we have estimated the model transition probability, then we can compute the code-length for K_t given $\hat{K}^{t-1}$ by

$$\ell(K_t|\hat{K}^{t-1}) = -\log P(K_t|\hat{K}^{t-1} : \hat{\alpha}_t). \quad (7)$$

Plugging Eq.(4) and Eq.(7) into Eq.(2) yields the following form of the sequential DMS criterion:

$$\begin{aligned} & L(X_t, Z_t, K_t|X^{t-1}, \hat{Z}^{t-1}, \hat{K}^{t-1}) \\ = \; & \ell_{\mathrm{RNML}}(X_t, Z_t|X^{t-1}, \hat{Z}^{t-1} : K_t \cdot \hat{K}^{t-1}; \gamma) + \ell(K_t|\hat{K}^{t-1}) \\ = \; & -\log \frac{f_{\mathrm{NML}}(X_t, Z_t; \hat{\eta}(X_t, Z_t), K_t)}{\sum_Z \int_{X \in Y'(\gamma)} f_{\mathrm{NML}}(X, Z; \hat{\eta}(X, Z) K_t) \mathrm{d}X} \\ & - \log P(K_t|\hat{K}^{t-1}; \hat{\alpha}_t). \end{aligned} \quad (8)$$

The sequential DMS algorithm outputs K_t and Z_t that minimize (8) at each time t. It is given in Algorithm 1.

2.2 Re-normalized Maximum Likelihood Code-length

Next, we show how to compute the RNML code-length (4) for a GMM as in the sequential DMS criterion (8).

Let $\mathbf{x}^n = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $\mathbf{x}_i = (x_{i1}, \ldots, x_{im})^\top$ $(i = 1, \ldots, n)$ be a given sequence where $\mathbf{x}_i$ is distributed according to a Gaussian distribution with the mean $\mu \in \mathbb{R}^m$ and the variance-covariance matrix $\Sigma \in \mathbb{R}^{m \times m}$ for a some positive integer m with density:

$$f(\mathbf{x}; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{m}{2}} |\Sigma|^{\frac{1}{2}}} \exp\Big\{ -\frac{1}{2}(\mathbf{x} - \mu)^\top \Sigma^{-1} (\mathbf{x} - \mu) \Big\}.$$

The NML distribution for a Gaussian distribution is given by

$$f_{\mathrm{NML}}(\mathbf{x}^n) \stackrel{\mathrm{def}}{=} \frac{f(\mathbf{x}^n; \hat{\mu}(\mathbf{x}^n), \hat{\Sigma}(\mathbf{x}^n))}{\int f(\mathbf{y}^n; \hat{\mu}(\mathbf{y}^n), \hat{\Sigma}(\mathbf{y}^n)) \mathrm{d}\mathbf{y}^n}. \quad (9)$$

The reason why we employ this code-length is that it is optimal in the sense that it attains Shtarkov's minimax criterion [13]. Notice here that the normalization term in Eq.(9) diverges. According to [5], we restrict the range of data so that the MLE lies in a bounded range specified by a parameter. Then the NML distribution is given as follows:

$$f_{\mathrm{NML}}(\mathbf{x}^n; R, \lambda_{\min}) \stackrel{\mathrm{def}}{=} \frac{f(\mathbf{x}^n; \hat{\mu}(\mathbf{x}^n), \hat{\Sigma}(\mathbf{x}^n))}{\mathcal{C}(R, \lambda_{\min})},$$

where

$$\begin{aligned} \mathcal{C}(R, \lambda_{\min}) &= \int_{Y(R, \lambda_{\min})} f(\mathbf{y}^n; \hat{\mu}(\mathbf{y}^n), \hat{\Sigma}(\mathbf{y}^n)) \mathrm{d}\mathbf{y}^n, \\ Y(R, \lambda_{\min}) &\stackrel{\mathrm{def}}{=} \{\mathbf{y}^n | \;\; ||\hat{\mu}(\mathbf{y}^n)||^2 \le R, \; \lambda_{\min}^{(j)} \le \hat{\lambda}_j(\mathbf{y}^n) \\ & \qquad (j = 1, \ldots, m), \; \mathbf{y}^n \in \mathcal{X}^n\}, \end{aligned} \quad (10)$$

Algorithm 1 The sequential DMS algorithm

STEP 1. At time $t = 1$, compute $\hat{Z}_1$ and $\hat{K}_1$ from X_1.
STEP 2. At time $t = 2, \ldots,$, compute $\hat{Z}_t$ and $\hat{K}_t$ as follows:
for $t \in \{2, 3, \ldots\}$ **do**
Consider the three cases of cluster changes: $\{-1, 0, +1\}$.
if $K_t^0 = \hat{K}_{t-1}$ **then**
· $Z_0 \leftarrow \hat{Z}_{t-1}$
· Run the EM algorithm with Z_0 being initial values to obtain Z_t^0.
· $L_{-1} \leftarrow L(X_t, Z_t^0, \hat{K}_{t-1}|X^{t-1}, \hat{Z}^{t-1}, \hat{K}^{t-1})$
else if $K_t^{-1} = \hat{K}_{t-1} - 1$ **then**
· $Z_0 \leftarrow \hat{Z}_{t-1}$
· From Z_0, pick up all the data such that $z_{i0} = K_{t-1}$ $(i = 1, \ldots, n)$ and assign them to other clusters randomly.
· Run the EM algorithm with Z_0 being initial values to obtain Z_t^{-1}.
· $L_0 \leftarrow L(X_t, Z_t^{-1}, \hat{K}_{t-1} - 1|X^{t-1}, \hat{Z}^{t-1}, \hat{K}^{t-1})$
else if $K_t^1 = \hat{K}_{t-1} + 1$ **then**
· $Z_0 \leftarrow \hat{Z}_{t-1}$
· Generate indices $i_1, \ldots, i_p$ $(p < n)$ randomly.
· $z_{j0} \leftarrow \hat{K}_{t-1} + 1$ $(j = i_1, \ldots, i_p)$
· Run the EM algorithm with Z_0 being initial values to obtain Z_t^1.
· $L_1 \leftarrow L(X_t, Z_t^1, \hat{K}_{t-1} + 1|X^{t-1}, \hat{Z}^{t-1}, \hat{K}^{t-1})$
end if
Compute $\hat{Z}_t$ and $\hat{K}_t$–the values after the change as follows:
· $change = \arg\min_c \; L_c$ $(c = -1, 0, 1)$
· $\hat{Z}_t \leftarrow Z_t^{change}$
· $\hat{K}_t \leftarrow K_t^{change}$
end for

where R and $\lambda_{\min} = (\lambda_{\min}^{(1)}, \ldots, \lambda_{\min}^{(m)})$ are parameters, and $\hat{\lambda}_j(\mathbf{y}^n)$ is the j-th largest eigenvalue of $\hat{\Sigma}(\mathbf{y}^n)$. If we let R and $\lambda_{\min}$ bounded, then the normalization term is also bounded.

Note here that the normalization term depends on the choice of the parameters R and $\lambda_{\min}$. Next, we consider the optimization of the NML code-length with respect to R and $\lambda_{\min}$. The terms including R and $\lambda_{\min}$ in the NML code-length are given by:

$$\frac{m}{2} \log R - \frac{m}{2} \sum_{j=1}^{m} \log \lambda_{\min}^{(j)}.$$

Considering the range of parameters (10), the MLE of R and $\lambda_{\min}$ are given as follows:

$$\begin{aligned} \hat{R}(\mathbf{y}^n) &= ||\hat{\mu}(\mathbf{y}^n)||^2, \\ \hat{\lambda}_{\min}^{(j)}(\mathbf{y}^n) &= \hat{\lambda}_j(\mathbf{y}^n) \; (j = 1, \ldots, m). \end{aligned}$$

We then introduce the hyper parameters $\gamma = (\lambda_1, \lambda_2, R_1, R_2)$ and define the renormalized maximum likelihood (RNML) distribution by

$$f_{\mathrm{RNML}}(\mathbf{x}^n; \gamma) = \frac{f_{\mathrm{NML}}(\mathbf{x}^n; \hat{R}(\mathbf{x}^n), \hat{\lambda}_{\min}(\mathbf{x}^n))}{\mathcal{C}(\gamma)},$$

where the normalization term is expanded as follows:

$$\begin{aligned} \mathcal{C}(\gamma) &= \int_{Y'(\gamma)} f_{\mathrm{NML}}(\mathbf{y}^n; \hat{R}(\mathbf{y}^n), \hat{\lambda}_{\min}(\mathbf{y}^n)) \mathrm{d}\mathbf{y}^n, \\ Y'(\gamma) &= \{\mathbf{y}^n | \; V(\sqrt{R_1}) \le V(\sqrt{\hat{R}(\mathbf{y}^n)}) \le V(\sqrt{R_2}), \end{aligned}$$

$$\lambda_1 \leq \hat{\lambda}^{(j)}_{\min}(\mathbf{y}^n) \leq \lambda_2 \ (j = 1, \ldots, m),\ \mathbf{y}^n \in \mathcal{X}^n\},$$

where $V(r) = 2\pi^{\frac{m}{2}} r^m / (m\Gamma(\frac{m}{2}))$, which denotes the volume of the m-dimensional ball with radius r.

Hirai and Yamanishi [6] proved that the RNML code-length for a GMM was further expanded as follows:

THEOREM 2.1. *[6] The RNML code-length of $\mathbf{x}^n$ relative to a GMM is expanded as follows:*

$$\begin{aligned}
&\ell_{\mathrm{RNML}}(\mathbf{x}^n, z^n; \gamma, K) \\
&\quad = -\log f(\mathbf{x}^n, z^n; K, \hat{\mu}(\mathbf{x}^n, z^n), \hat{\Sigma}(\mathbf{x}^n, z^n)) + \log \mathcal{C}_1(K, n) \\
&\qquad + \log \mathcal{C}_2(K, n) + \log B(\mathbf{x}^n, z^n) + K \log I(m, \gamma), \quad (11)
\end{aligned}$$

where

$$\mathcal{C}_1(K, n) = \sum_{h_1 + \cdots + h_K = n} \frac{n!}{h_1! \cdots h_K!} \prod_{k=1}^{K} \left(\frac{h_k}{n}\right)^{h_k}, \quad (12)$$

$$\mathcal{C}_2(K, n) = \sum_{h_1 + \cdots + h_K = n} \frac{n!}{h_1! \cdots h_K!} \prod_{k=1}^{K} \left(\frac{h_k}{n}\right)^{h_k} \cdot J(h_k), \quad (13)$$

$$B(\mathbf{x}^n, z^n) = \prod_{p=1}^{K} \frac{2^{m+1} \cdot ||\hat{\mu}_p(\mathbf{x}^n, z^n)||^m \cdot |\hat{\Sigma}_p(\mathbf{x}^n, z^n)|^{-\frac{m}{2}}}{m^{m+1}\Gamma(\frac{m}{2})},$$

$$I(m, \gamma) = \mathcal{C}(\gamma) = \left(\frac{m}{2}\right)^{m+1} \cdot \log \frac{R_2}{R_1} \cdot \left(\log \frac{\lambda_2}{\lambda_1}\right)^m,$$

$$J(h_k) = \left(\frac{h_k}{2\mathrm{e}}\right)^{m h_k} \cdot \frac{1}{\Gamma_m(\frac{h_k - 1}{2})}. \quad (14)$$

Here h_k denotes the number of data belonging to the k-th cluster, and $\hat{\mu}_p$ and $\hat{\Sigma}_p$ denote the MLE of the mean and the variance-covariance matrix of the p-th cluster.

Note that a straightforward computation of $\mathcal{C}_1(K, n)$ and $\mathcal{C}_2(K, n)$ as in Eq.(12) and Eq.(13) requires $\mathrm{O}(n^K)$ time. Below we give methods for efficient computation of $\mathcal{C}_1(K, n)$ and $\mathcal{C}_2(K, n)$. As for the computation of $\mathcal{C}_1(K, n)$, Kontkanen and Myllymäki [7] proved the following theorem:

THEOREM 2.2. *[7] $\mathcal{C}_1(K, n)$ satisfies the following recursive formula:*

$$\mathcal{C}_1(K + 2, n) = C_1(K + 1, n) + \frac{n}{K} C_1(K, n).$$

Hence $\mathcal{C}_1(K, n)$ is computed in $\mathrm{O}(n + K)$ time.

As for the computation of $\mathcal{C}_2(K, n)$, Hirai and Yamanishi [6] gave the following result:

THEOREM 2.3. *[6] $\mathcal{C}_2(K, n)$ satisfies the following recursive formula:*

$$\mathcal{C}_2(K+1, n) = \sum_{r_1 + r_2 = n} {}_nC_{r_1} \left(\frac{r_1}{n}\right)^{r_1} \left(\frac{r_2}{n}\right)^{r_2} C_2(K, r_1) J(r_2),$$

where $J(r_2)$ is as in Eq.(14). Hence $\mathcal{C}_2(K, n)$ is computed in $\mathrm{O}(n^2 K)$ time.

Combining all of the theorems as above, the RNML code-length of $\mathbf{x}^n$ relative to a GMM is computed in $\mathrm{O}(n^2 K)$ time. We employ Eq.(11) as the RNML code-length in Eq.(8).

3. EXPERIMENTAL RESULTS

We give results on empirical comparison of the sequential DMS algorithm with other methods using artificial data sets.

3.1 Comparison Methods

First we consider variants of the sequential DMS algorithm in which the RNML code-length, Schwarz's Bayesian information criterion (BIC) [12], and Akaike's information criterion (AIC) [1] are employed as criteria for selecting the optimal number of clusters, which we call these methods RNML, BIC, and AIC, respectively.

For the RNML code-length, the sequential DMS criterion is given by Eq.(8). If the RNML code-length is replaced with BIC, the sequential DMS criterion is given as follows:

$$\begin{aligned}
&L_{\mathrm{BIC}}(X_t, Z_t, K_t | X^{t-1}, \hat{Z}^{t-1}, \hat{K}^{t-1}) \\
&\quad = -\log f(X_t, Z_t; \hat{\theta}(X_t, Z_t), K_t) + \frac{1}{2} K_t \log n \\
&\qquad + \frac{m(m+3)K_t}{4} \sum_{k=1}^{K_t} \log h_k - \log P(K_t | \hat{K}^{t-1}),
\end{aligned}$$

where m is the dimension of each data. If the RNML code-length is replaced with AIC, the sequential DMS criterion is given as follows:

$$\begin{aligned}
&AIC(X_t, Z_t, K_t | X^{t-1}, \hat{Z}^{t-1}, \hat{K}^{t-1}) \\
&\quad = -\log f(X_t, Z_t; \hat{\theta}(X_t, Z_t), K_t) + \frac{1}{2} m(m+3)K + \frac{1}{2} K,
\end{aligned}$$

where the code-length for a model change is not added to AIC because AIC has no interpretation of a code-length. All of RNML, AIC, and BIC are obtained by changing criteria in Algorithm 1. We compared these algorithms.

We introduce three criteria to evaluate the algorithms

1. **AR** (accuracy rate): It is defined as the average rate of correctly estimating the true number of clusters over all time where the average is taken over all 100 trials. See Figure 3.2 for the illustration of the estimation of the number of clusters.
2. **IR** (identification rate): It is defined as the probability of correctly estimating the locations of change-points and the number of clusters before and after change-points over all 100 trials.
3. **FAR** (false alarm rate): It is defined as the rate of the number of false alarms over all detected change-points where the average is taken over all 100 trials.

3.2 Comparison of Criteria

We prepared two artificial data sets to compare RNML, AIC, and BIC in terms of the three indices as above.

3.2.1 Data Set 1

We generated Data Set 1 so that the number of clusters changed as follows:

$$K_t = \begin{cases} 3 & \text{if } 1 \leq t \leq 50, \\ 4 & \text{if } 51 \leq t \leq 100. \end{cases}$$

Algorithm 2 is the one which generated this data set. It created a new cluster at time 51.

For this data set, we conducted experiments 100 times by taking 100 different initial values. Figure 3.1 shows the average numbers of clusters and standard deviation per time for RNML, AIC, and BIC. Figure 3.2 shows **AR**s for RNML, AIC, and BIC. We observe that RNML was able to track changes of the number of clusters in a more stable way than AIC and BIC. Figure 3.3 shows **AR**s, **IR**s, and **FAR**s for RNML, AIC, and BIC. We observe that RNML was able to detect true change-points, and achieved significantly higher **AR** and **IR** and significantly less **FAR** than AIC and BIC.

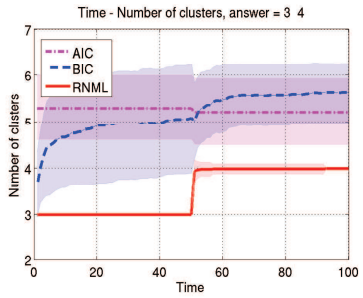

Figure 3.1: Data Set 1: Average Number of Clusters Over Time

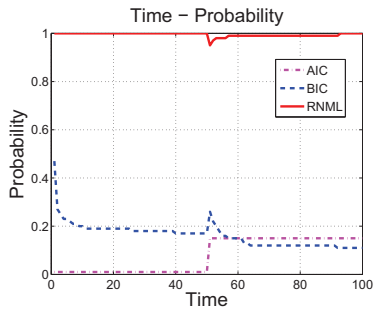

Figure 3.2: Data Set 1: Accuracy Rate Over Time

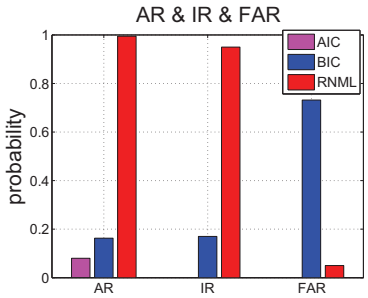

Figure 3.3: Data Set 1: AR, IR, FAR

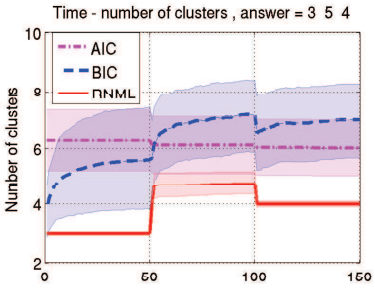

Figure 3.4: Data Set 2: Average Number of Clusters Over Time

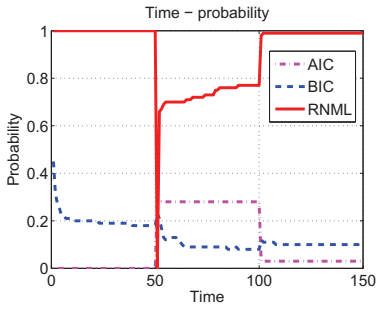

Figure 3.5: Data Set 2: Accuracy Rate Over Time

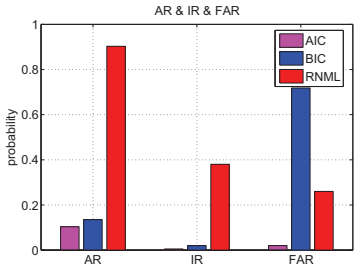

Figure 3.6: Data Set 2: AR, IR, FAR

Algorithm 2 Data Generation Algorithm

STEP 1. At time $t = 1$, generate X_1, Z_1.

STEP 2. At time $t = 2, \ldots, t_1 - 1$, $X_t \leftarrow X_1 + \epsilon, Z_t \leftarrow Z_1$. Here $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ and I is an identity matrix.

STEP 3. At change-points $t_1, \ldots, t_c$, generate data as follows:

for $j = 2$ to c **do**
  if $K_j = K_{j-1} + 1$ **then**
    at time t_j, choose N_+ data randomly, and assign them to a new cluster K_j.
    for $t = t_j + 1$ to $t_{j+1} - 1$ **do**
      $X_t \leftarrow X_{t_j} + \epsilon$
      $Z_t \leftarrow Z_{t_j}$
    end for
  else if $K_j = K_{j-1} - 1$ **then**
    at time t_j, choose data belonging to the cluster K_j, and assign them to other clusters randomly.
    for $t = t_j + 1$ to $t_{j+1} - 1$ **do**
      $X_t \leftarrow X_{t_j} + \epsilon$
      $Z_t \leftarrow Z_{t_j}$
    end for
  end if
end for

3.2.2 *Data Set 2*

We generated Data Set 2 so that the number of clusters changed as follows:

$$K_t = \begin{cases} 3 & \text{if } 1 \le t \le 50, \\ 5 & \text{if } 51 \le t \le 100, \\ 4 & \text{if } 101 \le t \le 150, \end{cases}$$

This data set was also generated by Algorithm 2. It created 2 new clusters at time 51 (the number of clusters increased from 3 to 5 at time $t = 51$ and an existing cluster disappeared at time 101).

Figures 3.4, 3.5, and 3.6 show the changes of the detected number of clusters and standard deviation, **AR**s per time, and **AR**s, **IR**s, and **FAR**s for RNML, AIC, and BIC.

We observe that RNML was able to identify the true numbers of clusters with higher probability and was able to detect change-points with significantly higher accuracy than AIC and BIC. Meanwhile, as the number of changes increased, the accuracy rates for BIC gradually decreased.

Note that RNML is designed assuming that the number of clusters changes by $\{-1, 0, +1\}$ from time t to $t+1$. We see from Figure 3.4 and 3.5 that even when the number of clusters changes by more than 1, say, 2 from time t to $t+1$, RNML was able to detect the changes step by step within an interval of $[t, t+c]$ for some small c ($c = 2$ in this experiment) for more than 66% data sets.

3.2.3 *Comparison with respect to KL-divergence*

We evaluated change detection accuracies for RNML, AIC and BIC by varying the Kullback-Leibler divergence (KLD) between the distributions before and after a change point. In general, the larger the KLD is, the more accurately change-points can be detected. We investigated how the detection accuracies varied as the KL-divergence varied. We generated Data Set 3 so that the number of clusters changed as follows:

$$K_t = \begin{cases} 3 & \text{if } 1 \le t \le 50, \\ 4 & \text{if } 51 \le t \le 100. \end{cases}$$

This data set was also generated by Algorithm 2. It created a new cluster at time 51, for which we took various mean values μ_{new}. The KLD between the GMMs before and after a change point was not strictly calculated. Hence we approximated it by the following formula:

$$KLD(f||g) = \sum_a \pi_a \log \frac{\sum_{a'} \pi_{a'} \exp(-D(f_a||f_{a'}))}{\sum_b \omega_b \exp(-D(f_a||g_b))}.$$

It is known from [4] that it gives a tight lower bound on the KLD between GMMs. Here f and g denote GMMs, f_a and g_b denote Gaussian components with indices a and b in f and g, and $D(f_a||g_b)$ denotes the KLD between f_a and g_b.

Figures 3.7–3.9 show **AR**s, **IR**s, and **FAR**s against the KLD, respectively. One plot corresponds to **AR** or **IR** or **FAR** for one mean value μ_{new} of a new cluster.

We observe that RNML was able to detect change-points with significantly higher accuracy than AIC/BIC in almost all plots. We further see that the larger the KLD between GMMs before and after the change-point was, the more accurately it was detected in terms of **AR**, **IR**, and **FAR**. This tendency turned out most clearly in RNML.

3.3 Comparison with Song and Wang's Alg.

We employ Song and Wang's algorithm [14], which we abbreviate as SW, for the comparison with our proposed method. SW is a hypothesis testing based algorithm, which is a different approach to clustering change detection from ours. Below we give its brief sketch. It first employs a GMM to conduct clustering of a newly input data, then makes statistical tests to determine whether the new cluster is identical to an old one or not. If it is, the new cluster is merged into the older one, otherwise it is recognized as a cluster which has newly emerged. SW is shown in Algorithm 3.

Algorithm 3 Algorithm: SW [14]

1: **INPUT** $g^N(x)$: a GMM at the latest time, $\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M}$: a newly input data sequence. Here N: data size at the latest time, M: data size of newly input data.
2: **OUTPUT** $g^{N+M}(x)$: a GMM.
3: Generate a GMM $a(x)$ from $\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M}$ where the mixture size K_a is determined by BIC.
4: **for** component k in $a(x)$ **do**
5: **for** component j in $g^N(x)$ **do**
6: Make a statistical test to determine whether Σ_k is identical to Σ_j.
7: If $\Sigma_k = \Sigma_j$, make a statistical test to determine whether μ_k is identical to μ_j or not.
8: **end for**
9: **end for**
10: Compare $g^N(x)$ with $a(x)$ to check if both of the mean and the variance-covariance matrix of one component in $g^N(x)$ are identical to those of another component in $a(x)$.
11: If they are identical, then merge a new cluster into the older one, and let the merged component be a component of $g^{N+M}(x)$.
12: Any component that is not merged into any old component be added into $g^{N+M}(x)$ as a new component.
13: Output a new GMM $g^{N+M}(x)$.

See [14] for the details of statistical tests for fitness of the means and the variance-covariance matrices.

It is assumed in SW that a set of data is newly input every time. In order to meet this setting, we conducted experiments by setting M to be $128, 256, 512, 1024$ in order. In SW the number of clusters is determined on the basis of BIC [12]. Hence we also consider a variant of SW such that BIC is replaced with the RNML criterion for the number of clusters selection criterion.

In summary, we compare the following three algorithms:

1. *Proposed* : Our proposed algorithm.
2. *SW-RNML* : A statistical-test based algorithm using on-line estimation of GMM with the RNML criterion (a variant of SW).
3. *SW-BIC* : A statistical-test based algorithm using on-line estimation of GMM with BIC (original SW).

For comparison, we used the performance indices: **AR**, **IR**, and **FAR**. Tables 3.1–3.3 show results for the three algorithms. Red numerical values indicate the winners.

We observe that our proposed algorithm was the winner in terms of all the indices: **AR**, **IR**, and **FAR** and that it significantly outperformed SW-BIC and SW-RNML. Comparing SW-BIC with SW-RNML, we see that the latter performed significantly better than the former in terms of **AR** and **IR**, while they were comparable in terms of **FAR** when the data size was small. In general, all of the indices for SW-BIC and SW-RNML got better as the data size increased. Meanwhile, our proposed algorithm performed well without depending on the data size so much.

4. MARKET ANALYSIS APPLICATION

4.1 Transaction Data Set

We present an application of clustering change detection to market analysis. The experiment was conducted in corporation with HAKUHODO, Inc. We employed a real data set provided by MACROMILL, Inc. It is specified as follows:

1. *Period* : November 1st 2010 to January 1st 2011.
2. *Number of customers* : 3185 customers.
3. *Data specification* : The data set consists of a number of customers' beer purchase transactions. Each customer's record is specified by a 14-dimensional feature vector, each component of which shows consumption for a certain beer. Brands of beer includes: {Beer-A, Beer-B, Premium-A, Premium-B, Beer-C, Beer-D, Third-A, Third-B, Third-C, Third-D, LM-Beer-A, Off-A, Off-B, Off-C}. Premium-A, B are brands of relatively expensive beer. Third-A,B,C,D are brands of what we call the "third kind" of beer, i.e., beer-taste alcoholic drink. Off-A,B,C are brands of what we call "calorie off" type of beer. LM-Beer-A are beer-type alcoholic drink with relatively low rate of barley.

We constructed a sequence of customers' feature vectors as follows: A time unit is a day. At each time (day) t $(= \tau, \ldots, T)$, we denote the feature vector of the i-th customer as $\mathbf{x}_{it} = (x_{it,1}, \ldots, x_{it,d}) \in \mathbb{R}^d$ where $d = 14$. Each $\mathbf{x}_{it,j}$ is the i-th customer's consumption of the j-th brand beer from time $t - \tau + 1$ to t. We denote the set of data at time t as $X_t = (\mathbf{x}_{1t}, \ldots, \mathbf{x}_{n_t t})$, where n_t is the number of customers who purchased the products from time $t - \tau + 1$ to t.

4.2 Market Analysis

We evaluated the clustering change detection algorithms for this data set. Figures 4.1–4.3, and Tables 4.1–4.3 show the results at $\tau = 14$. Numerical values in Tables 4.1 and 4.2 show the average values of consumption where the average is

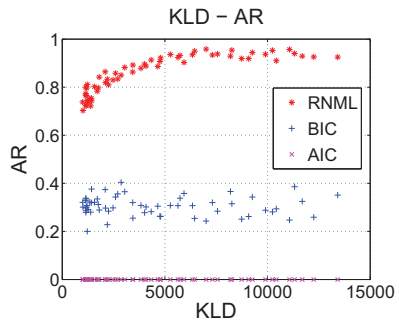

Figure 3.7: AR w.r.t. KLD

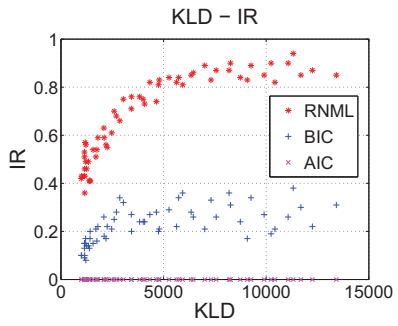

Figure 3.8: IR w.r.t. KLD

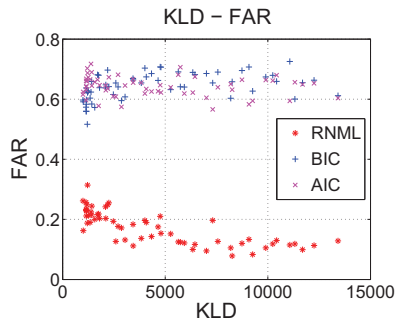

Figure 3.9: FAR w.r.t. KLD

Table 3.1: AR with Various Sample Size in Each Time

Algo. \ M	128	256	512	1024
Proposed	**0.972**	**0.997**	**0.988**	**0.975**
SW-RNML	0.027	0.238	0.369	0.750
SW-BIC	0.005	0.009	0.019	0.125

Table 3.2: IR with Various Sample Size in Each Time

Algo. \ M	128	256	512	1024
Proposed	**1.000**	**1.000**	**0.950**	**0.950**
SW-RNML	0.000	0.100	0.300	0.700
SW-BIC	0.000	0.000	0.000	0.050

Table 3.3: FAR with Various Sample Size in Each Time

Algo. \ M	128	256	512	1024
Proposed	**0.158**	**0.025**	**0.050**	**0.000**
SW-RNML	0.944	0.644	0.503	0.208
SW-BIC	0.964	0.930	0.841	0.550

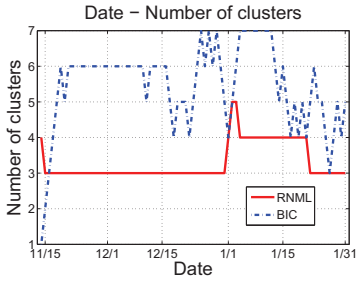

Figure 4.1: Changes of # of Clusters for RNML vs BIC

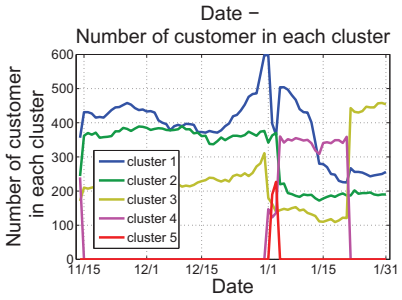

Figure 4.2: Changes of # of Customers within Individual Clusters

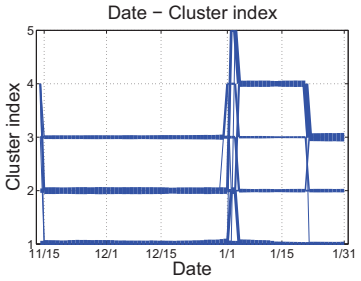

Figure 4.3: Changes of # of Customers among Clusters

taken over all customers within an individual cluster. Figure 4.1 shows the changes of the number of clusters per a day starting from Nov.14th, 2010.

We first compared our proposed algorithm, the sequential DMS with the RNML criterion, with that using BIC. Figure 4.1 shows the results of change detection from Nov. 14th on 2010 to Jan. 31st 2011. We see from Figure 4.1 that the sequential DMS with RNML was able to detect clustering changes that corresponded to the emergence and disappearance of the year's ending demand. Meanwhile, the sequential DMS with BIC detected much more change-points. They included those which were not necessarily related to the year's ending demand. It was so sensitive to irregularities of data that it produced too many change-points. Here is a list of other figures showing the results.

- Figure 4.2: It shows how the number of customers changed over time for individual clusters.
- Figure 4.3: It shows how many customers moved from clusters to clusters. The bolder the line is, the more customers moved. We skipped drawing for the movement of customers when they were not more than 20.

The DMS with RNML detected four change points: Nov. 15th 2010, Jan. 1st 2011, Jan. 2nd 2011, Jan. 4th, and Jan. 22nd 2011. For example, as for the clustering change from Dec. 31st 2010 to Jan. 2nd 2011, Table 4.1 shows the average values of consumption for each beer within individual clusters before the change, while Table 4.2 shows those after the change. In Tables 4.1 and 4.2, $c_1, c_2, \ldots$ denote cluster 1, cluster 2, ..., respectively.

Table 4.3 shows how customers moved from clusters to clusters from Dec. 31st 2010 to Jan. 2nd 2011. In Table 4.3, each row shows a cluster on Dec. 31st 2010 while each column shows a cluster on Jan. 2nd 2011. For example, it is shown that 139 customers moved from the old cluster 1 on Dec. 31st 2010 to the new cluster 5 on Jan. 1st 2011.

We see from Tables 4.1–4.3 that most of customers belonging to the old cluster 3, which had the largest consumption on Dec. 31st, moved into the new clusters 3 and 4.

The cluster 5 newly emerged on Jan. 1st. This cluster had large consumption for Beer A and Third Beer. Most of customers belonging to this cluster came from the old clusters 1 and 2, where the customers belonging to the old cluster 1 had relatively large consumption of Beer(A,B) and Premium-beer(A,B) while the customers belonging to the old cluster 2 had relatively large consumption of other types of beer (Third Beer, LM-Beer, Off-Beer). We see that many of customers belonging to the old clusters 1 and 2 changed their patterns to purchase Beer A and Third Beer at the years' end. Through this analysis, it has turned out that the clustering change detection leads to the understanding of how customer groups changed and how customers moved from clusters to clusters. It was validated by domain experts.

Table 4.1: Clustering Structure on Dec. 31st

Cluster	c 1	c 2	c 3
Beer-A	184	0	117
Beer-B	91	0	95
Premium-A	108	0	80
Premium-B	113	0	43
Beer-C	0	0	126
Beer-D	0	0	140
Third-A	93	41	43
Third-B	0	198	121
Third-C	0	303	103
Third-D	0	120	182
LM Beer-A	0	75	48
Off-A	0	0	157
Off-B	0	114	34
Off-C	0	0	83
Total Volume	589	852	1373
# Customers	598	376	311

Table 4.2: Clustering Structure on Jan. 1st

Cluster	c 1	c 2	c 3	c 4	c 5
Beer-A	84	0	50	131	229
Beer-B	123	0	0	248	0
Premium-A	153	0	73	174	0
Premium-B	176	0	0	105	0
Beer-C	0	0	122	146	0
Beer-D	0	0	192	72	0
Third-A	101	131	0	130	0
Third-B	0	34	0	406	131
Third-C	0	107	46	112	236
Third-D	0	202	0	431	0
LM-Beer A	0	107	0	87	0
Off-A	0	0	138	169	0
Off-B	0	215	0	74	0
Off-C	0	0	83	61	0
Total Volume	637	796	705	2348	596
# Customers	397	190	162	123	363

5. CONCLUDING REMARKS

We have considered the clustering change detection issue. In it we have proposed the sequential DMS algorithm that sequentially tracks changes of clustering structures. The key ideas of the algorithm are; 1) extending DMS (dynamic model selection) into a sequential clustering scenario, and 2) the use of the RNML (renormalized maximum likelihood) code-length in the DMS criterion. The proposed algorithm enables us to deal with merging, splitting, emergence, disappearance, etc. of clusters from a unifying view of the MDL principle. We have empirically demonstrated using artificial data sets that our algorithm is able to detect cluster changes much more accurately than AIC/BIC-based methods and the existing statistical-test based method. We have used real customers' transaction data sets to demonstrate the validity of our algorithm in market analysis.

Acknowledgements

This work was partially supported by MEXT KAKENHI 23240019, Aihara Project, the FIRST program from JSPS, initiated by CSTP, HAKUHODO Inc., MACROMILL, Inc., NTT Corporation, and Microsoft Corporation(CORE6 Project).

Table 4.3: Movement of Customers from Dec. 31st to Jan.1st

cluster	no	c 1	c 2	c 3	c 4	c 5
no purchase	0	22	8	13	3	29
c 1	61	355	26	9	8	139
c 2	45	4	152	1	8	166
c 3	19	16	4	139	104	29

6. REFERENCES

[1] H. Akaike. A new look at the statistical model identification. *IEEE Trans. on Automatic Control*, 19(6):716–723, Dec. 1974.

[2] D.Chakrabrti, R.Kumar. Evolutionary clustering. *Proc.KDD06*, pp:554-560. 2006.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em. *J. Royal Staitst. Soc. B*, 39:1–38, 1977.

[4] J. Hershey and P. Olsem. Approximating the kullback Leibler divergence between Gaussian mixture models. *Proc. of ICASSP*, 4:317–320, 2007.

[5] S. Hirai and K. Yamanishi. Efficient computation of normalized maximum likelihood coding for Gaussian mixtures with its applications to optimal clustering. *Proc. of IEEE ISIT*, pp.1031–1035, 2011.

[6] S. Hirai and K. Yamanishi. Normalized maximum likelihood coding for exponential family with its applications to optimal clustering. *arXiv:1205.3549*, 2012.

[7] P. Kontkanen and P. Myllymäki. A linear time algorithm for computing the multinomial stochastic complexity. *Inf. Proc. Letters*, 103:227–233, 2007.

[8] Z. G. Krempl and M.Spiliopoulou. Online clustering of high-dimensional trajectories under concept drift. *Proc. ECML-PKDD2011, Part II*, pp. 261–276, 2011.

[9] R. E. Krichevsky and V. K. Trofimov. The performance of universal encoding. *IEEE Trans. Inf. Theory*, 27:199–207, 1981.

[10] J. Rissanen. *Stochastic Complexity in Statistical Inquiries*. World Scientific, 1989.

[11] M. Sato. Online model selection based on the variational bayes. *NC*, 13:1649–1681, 2001.

[12] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics 6 (2)*, pp. 461–464, 1978.

[13] Y. M. Shtarkov. Universal sequential coding of single messages. *Problems of Information Transmission*, 23(3):3–17, 1987.

[14] M. Song and H. Wang. Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering. *Intelligent Computing: Theory and Application*, 2005.

[15] J. Sun, S. Papadimitriou, P. S. Yu, and C. Faloutsos. Graphscope: Parameter-free mining of large time evolving graphs. *Proc. KDD07*, pp: 687–696, 2007.

[16] K. Yamanishi and Y. Maruyama. Dynamic syslog mining for network failure monitoring. *Proc. of KDD2005*, 499–508, 2005.

[17] K. Yamanishi and Y. Maruyama. Dynamic model selection with its applications to novelty detection. *IEEE Trans. on Inf. Theory*, 53(6):2180–2189, 2007.