Chapter 5

Information-Theoretic Clustering and Algorithms

Toshio Uchiyama

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/66588

Abstract

Clustering is the task of partitioning objects into clusters on the basis of certain criteria so that objects in the same cluster are similar. Many clustering methods have been proposed in a number of decades. Since clustering results depend on criteria and algorithms, appropriate selection of them is an essential problem. Recently, large sets of users' behavior logs and text documents are common. These are often presented as high-dimensional and sparse vectors. This chapter introduces information-theoretic clustering (ITC), which is appropriate and useful to analyze such a high-dimensional data, from both theoretical and experimental side. Theoretically, the criterion, generative models, and novel algorithms are shown. Experimentally, it shows the effectiveness and usefulness of ITC for text analysis as an important example.

Keywords: information-theoretic clustering, competitive learning, Kullback-Leibler divergence, Jensen-Shannon divergence, clustering algorithm, text analysis

1. Introduction

Clustering is the task of partitioning objects into clusters on the basis of certain criteria so that objects in the same cluster are similar. It is a fundamental procedure to analyze data [1, 2].

Clustering is *unsupervised* and different from *supervised* classification. In supervised classification, we have a set of *labeled* data (belong to predefined classes), train a classifier using the labeled data (*training set*), and judge which class a new object belongs to by the classifier. In the case of clustering, we find meaningful clusters without using any labeled data and group a given collection of unlabeled data into them. Clustering can also help us to find meaningful classes (labels) for supervised classification. Since it is more difficult to prepare the training set for larger data sets, recently unsupervised analysis of data such as clustering becomes more important.

For example, **Table 1** *user-item matrix* shows which item a user bought. When considering the data as a set of feature vectors for users, we can find a lot of types of users' behavior by

	item1	item2	item3	item4	item5
User1	3	0	0	5	0
User2	0	1	2	0	0
User3	1	0	0	0	2
User4	0	0	1	0	0

$$\Rightarrow \begin{pmatrix} 3 & 0 & 0 & 5 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Table 1. Consumption behavior of users.

	Document1	Document2	Document3	Document4	Document5
Word1	3	0	0	5	0
Word2	0	1	2	0	0
Word3	1	0	0	0	2
Word4	0	0	1	0	0

Table 2. Word frequencies in documents (bag-of-words feature representation).

clustering. It is also possible to analyze data as a set of feature vectors for items. From *word-document matrix* in **Table 2**, both document clusters and word clusters could be extracted.

Many clustering methods have been proposed in a number of decades. Those include k-means algorithm [3], competitive learning [4], spherical clustering [5], spectral clustering [6], and maximum margin clustering [7]. Since clustering results depend on criteria and algorithms, appropriate selection of them is an essential problem. Large sets of users' behavior logs and text documents (as shown in **Tables 1** and **2**) are common recently. These are often presented as high-dimensional and sparse vectors. This chapter introduces information-theoretic clustering [8] and algorithms that are appropriate and useful to analyze such a high-dimensional data.

Information-theoretic clustering (ITC) uses Kullback-Leibler divergence and Jensen-Shannon divergence to determine its criterion, while k-means algorithm uses the sum of squared error as criterion. This chapter explains ITC by contrasting these two clustering techniques (criteria and algorithms), because there are a number of interesting similarities between them. There exists difficulty in algorithms for ITC. We explain the details of it and propose novel algorithms to overcome.

Experimental results for text data sets are presented to show the effectiveness and usefulness of ITC and novel algorithms for it. In experiments, maximum margin clustering and spherical clustering are used to compare. We also provide the evidence to support the effectiveness of ITC by detailed analysis of clustering results.

2. The sum-of-squared-error criterion and algorithms

Given a set of M-dimensional input vectors $\mathcal{X} = \{\boldsymbol{x}^i | \boldsymbol{x}^i \in R^M, i = 1,\ldots,N\}$ where N is the number of vectors, clustering is the task of assigning each input vector $\boldsymbol{x}^i$ a cluster label $k(k = 1,\ldots,K)$ to

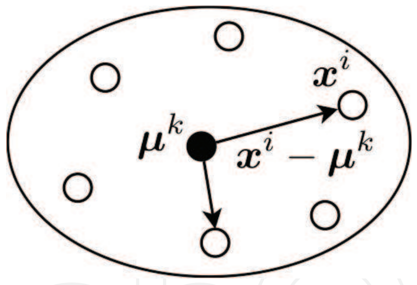

Figure 1. Input vectors and the mean vector in C^k.

partition them into K clusters $\mathcal{C} = \{C^1,\ldots,C^K\}$. The *sum-of-squared-error criterion* [9] is a simple and widely used criterion for clustering.

Let $\boldsymbol{\mu}^k$ be the mean of the input vectors $\boldsymbol{x}^i$ which belong to the cluster C^k (see **Figure 1**). Then, the error in C^k is the sum of squared lengths of the differential (= "error") vectors $\|\boldsymbol{x}^i - \boldsymbol{\mu}^k\|^2$ and the sum-of-squared-error criterion about all clusters (*within-cluster* sum of squares) is defined by

$$J_W = \sum_{k=1}^{K} \sum_{\boldsymbol{x}^i \in C^k} \|\boldsymbol{x}^i - \boldsymbol{\mu}^k\|^2. \tag{1}$$

J_W is the objective function (criterion) to be minimized in clustering based on this criterion.

Also, we define the sum of squares of *between-cluster* J_B and *total* J_T as

$$J_B = \sum_{k=1}^{K} N_k \|\boldsymbol{\mu}^k - \boldsymbol{\mu}\|^2, \; J_T = \sum_{i=1}^{N} \|\boldsymbol{x}^i - \boldsymbol{\mu}\|^2, \tag{2}$$

respectively, where N_k is the number of input vectors x^i in C^k (i.e., $N = \sum_{k=1}^{K} N_k$) and

$$\boldsymbol{\mu}^k = \frac{1}{N_k} \sum_{\boldsymbol{x}^i \in C^k} \boldsymbol{x}^i, \; \boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}^i. \tag{3}$$

It follows from these definitions that the total sum of squares is the sum of the within-cluster sum of squares and the between-cluster sum of squares:

$$J_T = J_W + J_B. \tag{4}$$

Since the mean of the all input vectors $\boldsymbol{\mu}$ is derived from $\mathcal{X} = \{\boldsymbol{x}^1,\ldots,\boldsymbol{x}^N\}$ [see Eq. (3)], J_T does not depend on clusters $\mathcal{C}$ [see Eq. (2)] and is constant for the given input vectors $\mathcal{X}$. Therefore, minimization of J_W is equivalent to maximization of J_B. In this sense, clustering based on minimizing this criterion J_W works to find separable clusters each other.

2.1. Generative model

In the background of the clustering based on the objective function (criterion) J_W, there exists assumption of Gaussian distribution about input vectors [10].

Suppose that there are clusters $C^k(k = 1,\dots,K)$, which generates input vectors by the conditional probability density function:

$$p(\boldsymbol{x}^i|\boldsymbol{x}^i \in C^k) = \frac{1}{(2\pi\sigma_k^2)^{M/2}} \exp\left(-\frac{\|\boldsymbol{x}^i - \boldsymbol{\mu}^k\|^2}{2\sigma_k^2}\right), \tag{5}$$

where σ_k is a standard deviation of the cluster C^k and M is the number of dimension of $\boldsymbol{x}^i$. In followings, we assume that σ_k is constant value σ for all clusters $C^k(k = 1,\dots,K)$. Considering independence of each generation, joint probability density function for the input vectors $\mathcal{X}$ becomes

$$p(\mathcal{X}|\mathcal{C}) = \prod_{k=1}^{K} \prod_{\boldsymbol{x}^i \in C^k} \frac{1}{(2\pi\sigma^2)^{M/2}} \exp\left(-\frac{\|\boldsymbol{x}^i - \boldsymbol{\mu}^k\|^2}{2\sigma^2}\right), \tag{6}$$

where $\mathcal{C}$ indicate cluster information that specifies which input vector $\boldsymbol{x}^i$ belongs to cluster C^k. Taking the logarithm of Eq. (6) yields

$$\ln p(\mathcal{X}|\mathcal{C}) = -\frac{NM}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{k=1}^{K}\sum_{\boldsymbol{x}^i \in C^k}\|\boldsymbol{x}^i - \boldsymbol{\mu}^k\|^2. \tag{7}$$

Since σ is constant, the maximization of Eq. (7) is equivalent to the minimization of

$$\sum_{k=1}^{K}\sum_{\boldsymbol{x}^i \in C^k}\|\boldsymbol{x}^i - \boldsymbol{\mu}^k\|^2. \tag{8}$$

which is nothing more or less than the objective function (criterion) J_W. Therefore, under the assumption of Gaussian distribution about input vectors, clustering based on Eq. (8) works to find the most probable solution $\mathcal{C}$.

2.2. Algorithms

2.2.1. k-means algorithm

k-means [3, 11] is well-known algorithm for clustering based on the sum-of-squared-error criterion. Main idea of this algorithm is as follows. In the objective function J_W (1), error for vector $\boldsymbol{x}$ is calculated by $\|\boldsymbol{x} - \boldsymbol{\mu}^k\|^2$ where $\boldsymbol{\mu}^k$ is the mean of cluster C^k to which $\boldsymbol{x}$ belongs. If $\|\boldsymbol{x} - \boldsymbol{\mu}^t\|^2 < \|\boldsymbol{x} - \boldsymbol{\mu}^k\|^2$, changing the cluster from C^k to C^t can reduce the objective function J_W.

We introduce **weight vector** $\boldsymbol{w}^k(k = 1,\dots,K)$ ($\mathcal{W}$) that represent cluster C^k to implement the idea mentioned above. The weight vector $\boldsymbol{w}^k$ involves mean vector $\boldsymbol{\mu}^k$ and prototype vector of cluster C^k. As illustrated in **Figure 2**, the idea of *k-means* is alternative repetition of two steps "(a) Update weights" (calculating mean $\boldsymbol{\mu}^k$ as weight vector $\boldsymbol{w}^k$) and "(b) Update clusters" (allocating input vector $\boldsymbol{x}^i$ to a cluster C^k on the basis of minimum length from weight vectors $\boldsymbol{w}^k$). Note that **Figure 2b** is a Voronoi tessellation determined by weight vectors $\boldsymbol{w}^k$, which are usually called *prototype vector* in this context.

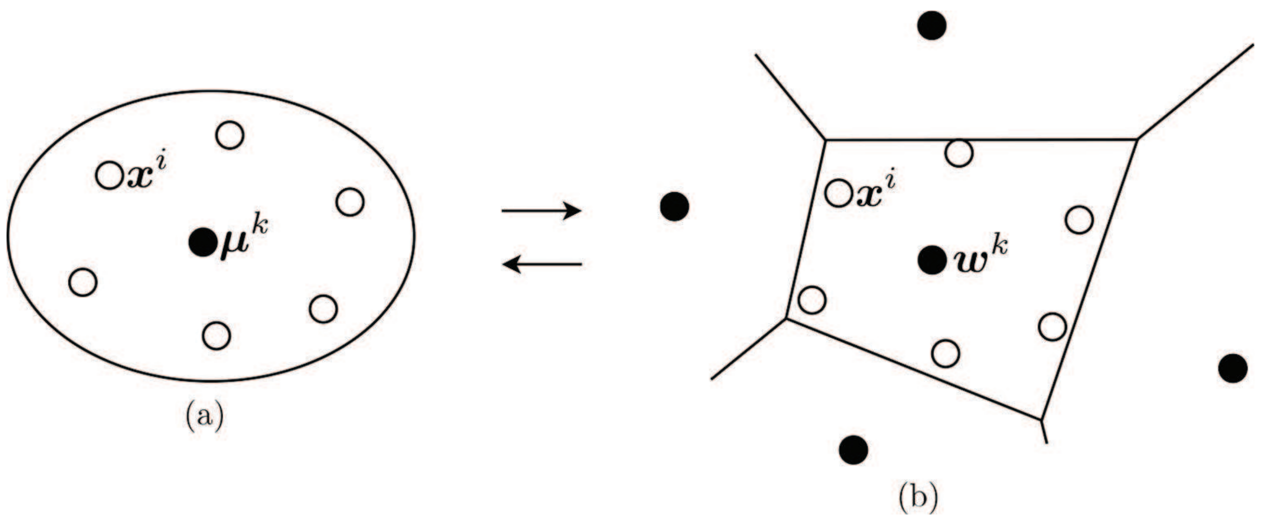

Figure 2. Two steps in k-means algorithm. (a) Update weights. (b) Update clusters.

Figure 3a is a flow chart of k-means algorithm to which processes of initialization and termination are added. As a matter of fact, clustering is closely related to vector quantization. Vector quantization means mapping input vectors to a codebook that is a set of weight vectors (prototype vectors). When using quantization error E_Q:

$$E_Q = \sum_{i=1}^{N} \min_k \|\boldsymbol{x}^i - \boldsymbol{w}^k\|^2, \tag{9}$$

clusters $\mathcal{C}$ determined by a local optimal solution of vector quantization $\mathcal{W}$ is a local optimal solution of clustering problem [12]. In this sense, clustering can be replaced by vector quantization and vice versa. We can write a flow chart for vector quantization as **Figure 3b**, but we also find this chart (b) as k-means algorithm. Furthermore, *LBG algorithm* [13], which is well known for vector quantization, is based on an approach of Lloyd [3] (one of original papers for k-means algorithm). These facts show a close relationship between clustering and vector quantization.

Initialization is important, because k-means algorithm converges to a local optimal solution which depends on an initial condition (a set of weights or clusters). If we initialize weights $\mathcal{W}$ by randomly selecting them from input vectors, it may converge to a very bad local optimal solution with high probability. *Random labeling* that randomly assigns cluster labels $\mathcal{C}$ to input vectors may lead to better solutions than random selection of weights. The initialization *Random labeling* can also be used for charts (b) and (c) in **Figure 3** by replacing “Initialize weights” step to “Initialize clusters” and “Update weights” steps. For directly initializing weights, splitting algorithm [13] and *k-means* ++ [14] were known.

2.2.2. Competitive learning

Competitive learning [4, 11] is a learning method for vector quantization and also utilized for clustering. While k-means algorithm updates all weights $\mathcal{W}$ by batch processing, competitive learning updates one weight $\boldsymbol{w}$ at a time to reduce a part of the quantization error Q_E (see **Figure 3c**) as

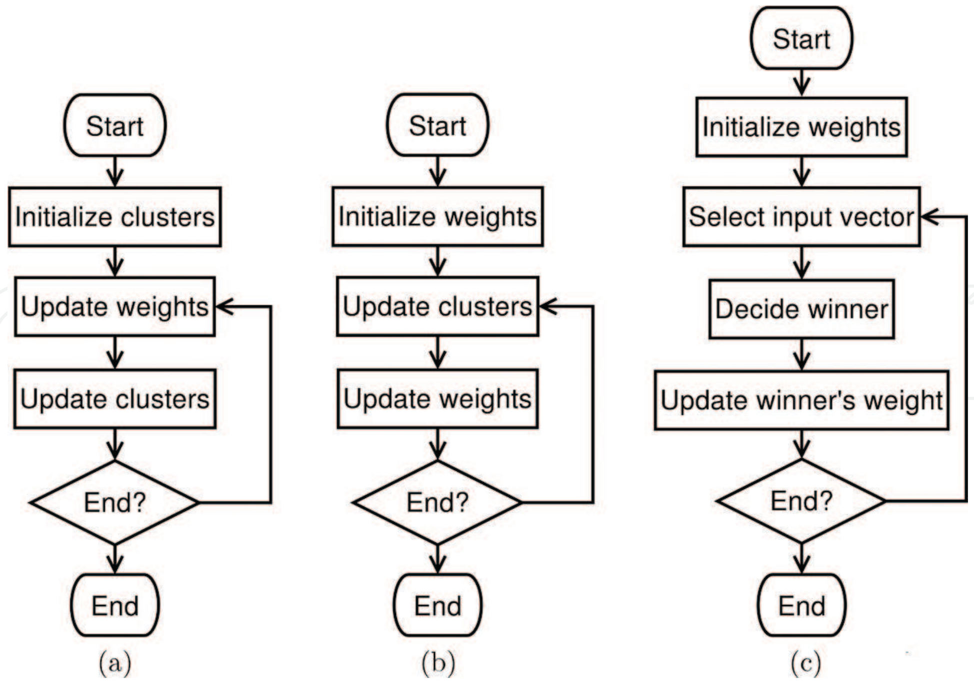

Figure 3. Flow charts of algorithms based on sum-of-squared-error criterion. (a) k-means1, (b) k-means2, and (c) competitive learning.

1. Select one input vector $\boldsymbol{x}$ randomly from $\mathcal{X}$.
2. Decide a winner $\boldsymbol{w}^c$ from $\mathcal{W}$ by

$$c = \arg\min_k \|\boldsymbol{x} - \boldsymbol{w}^k\|^2 \ \ \text{(If there are several candidates, choose the smallest k).} \tag{10}$$

3. Update the winner's weight $\boldsymbol{w}^c$ as

$$\boldsymbol{w}^c \leftarrow (1-\gamma)\boldsymbol{w}^c + \gamma\boldsymbol{x}, \tag{11}$$

where γ is a given learning rate (e.g., 0.01–0.1).

Though the *winner-take-all* update in Step 3 (**Figure 4**) that reduces partial error $\|\boldsymbol{x} - \boldsymbol{w}^c\|^2$ in steepest direction does not always reduce the total quantization error E_Q, repetition of the update can reduce E_Q on the basis of **stochastic gradient decent** method [15, 16]. For termination condition, maximum number of times of iteration N_r (*the number of maximum repetitions*) can be used. After termination, the step of deciding clusters $\mathcal{C}$ like **Figure 2b** is required for clustering purpose.

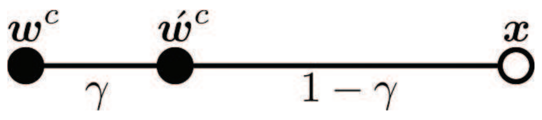

Figure 4. Update of the winner's weight in competitive learning.

Against natural expectations, competitive learning outperforms k-means without any contrivance in most cases. Furthermore, information obtained in learning process allows us to improve its performance. Splitting rule [12] utilizes the number of each weight $\boldsymbol{w}$ wins to estimate density around it. As **Figure 5a**, **b** shows, higher density of input vectors around makes the weight vector $\boldsymbol{w}^a$ win more frequently than $\boldsymbol{w}^b$.

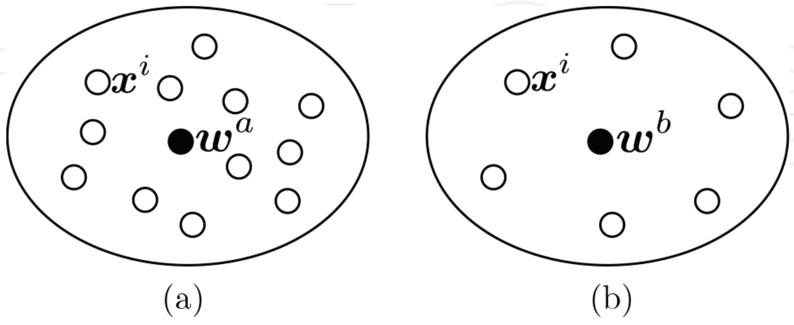

Figure 5. Density of input vectors around a weight vector.

Splitting rule in competitive learning [12] aims to overcome the problem of discrepancy between distribution of input vectors $\mathcal{X}$ and that of weight vectors $\mathcal{W}$. The discrepancy causes a few weight vectors $\boldsymbol{w}$ monopolize $\mathcal{X}$ and leads to a solution of very poor quality, but it is impossible to figure out the distribution of input vectors beforehand. Accordingly, this splitting rule distributes weight vectors $\boldsymbol{w}$ in learning process as

1. One weight vector $\boldsymbol{w}^1$ with a variable τ^1 is set. τ^1 denotes how many times weight vector $\boldsymbol{w}^1$ wins and is initialized to 0.
2. Select one input vector $\boldsymbol{x}$, decide winner $\boldsymbol{w}^c$, and update the winner's weight $\boldsymbol{w}^c$.
3. Add 1 to τ^c. If $\tau^c = \theta$ and the current number of weights K' is less than K, generate a new weight vector $\boldsymbol{w}$ which is the same as the winner $\boldsymbol{w}^c$ and clear τ of both to 0, where θ is the threshold of times for splitting.
4. Repeat 2 and 3 until termination condition is true.

3. Information-theoretic clustering and algorithms

Information-theoretic clustering (ITC) [8] is closely related to works about **distributional clustering** [17–19] and uses Kullback-Leibler divergence and Jensen-Shannon divergence to determine its criterion. Though there exists difficulty in algorithms and effectiveness for high-dimensional count data (e.g., text data), its definition and properties are similar to those of the sum-of-squared-error criterion. The main contributions of this chapter are to present the technique to overcome the difficulty and effectiveness of ITC.

Let $\mathcal{X} = \{\boldsymbol{x}^i | \boldsymbol{x}^i \in R_+^M, i = 1, \ldots, N\}$ be a set of M-dimensional input vectors (N denote the number of input vectors), where elements of vectors $\boldsymbol{x}$ are nonnegative real numbers. We define a l^1-norm of input vector $t^i (= \sum_m |x^i_m|)$, normalized input vectors $\boldsymbol{p}^i = \boldsymbol{x}^i / t^i$, and an input probability distribution P^i whose mth random variable takes the mth element of $\boldsymbol{p}^i (= p^i_m)$. Let $\mathcal{P} = \{P^1, \ldots, P^N\}$ be a set of input distributions (input data).

Suppose that we assign each distribution P^i a cluster label $k (k = 1, \ldots, K)$ to partition them into K clusters $\mathcal{C} = \{C^1, \ldots, C^K\}$.

Let $\overline{P}^k$ be the distributions on the mean of input data P^i which belong to the cluster C^k (see **Figure 6**). Then, the generalized Jensen-Shannon (JS) divergence to be minimized in C^k is defined by

$$D_{\mathrm{JS}}(\{P^i | P^i \in C^k\}) = \sum_{P^i \in C^k} \pi^i D_{\mathrm{KL}}(P^i \| \overline{P}^k), \ \overline{P}^k = \sum_{P^i \in C^k} \pi^i P^i, \tag{12}$$

where N_k is the number of distributions P^i in cluster C^k (i.e., $N = \sum_{k=1}^K N_k$), $D_{\mathrm{KL}}(P^i \| \overline{P}^k)$ is the Kullback-Leibler (KL) divergence to the mean distribution $\overline{P}^k$ from P^i, and π^i is the probability of P^i ($\sum_{P^i \in C^k} \pi^i = 1$). Here $P^i = 1/N_k$. Then, we define *within-cluster* JS divergence JS_W which considers all clusters $C^k (k = 1, \ldots, K)$ as

$$JS_W = \sum_{k=1}^K \frac{N_k}{N} D_{\mathrm{JS}}(\{P^i | P^i \in C^k\}) \tag{13}$$

$$= \frac{1}{N} \sum_{k=1}^K \sum_{P^i \in C^k} D_{\mathrm{KL}}(P^i \| \overline{P}^k) \tag{14}$$

$$= \frac{1}{N} \sum_{k=1}^K \sum_{P^i \in C^k} \sum_{m=1}^M p^i_m \log \frac{p^i_m}{\overline{p}^k_m} = \frac{1}{N} \sum_{k=1}^K \sum_{P^i \in C^k} \sum_{m=1}^M (p^i_m \log p^i_m - p^i_m \log \overline{p}^k_m). \tag{15}$$

The *within-cluster* JS divergence JS_W is the objective function (criterion) of **information-theoretic clustering (ITC)** to be minimized [8]. We also define JS divergence of *between-cluster* JS_B and *total* JS_T as

$$JS_B = D_{\mathrm{JS}}(\{\overline{P}^k | k = 1, \ldots, K\}) = \sum_{k=1}^K \pi^k D_{\mathrm{KL}}(\overline{P}^k \| \overline{P}), \ \pi^k = N_k / N, \tag{16}$$

$$JS_T = D_{\mathrm{JS}}(\{P^i | i = 1, \ldots, N\}) = \sum_{i=1}^N \pi^i D_{\mathrm{KL}}(P^i \| \overline{P}), \ \pi^i = 1/N, \tag{17}$$

where $\overline{P} = \sum_{i=1}^N \pi^i P^i = 1/N \sum_{i=1}^N P^i$ is the distribution on the mean of all input data. It follows from these definitions that the total JS divergence is the sum of the within-cluster JS divergence and the between-cluster JS divergence [8]:

$$JS_T = JS_W + JS_B. \tag{18}$$

Since JS_T are constant for given input distributions $\mathcal{P}$, minimization of JS_W is equivalent to maximization of JS_B. In this sense, clustering based on minimizing this criterion JS_W works to find separable clusters each other.

The definition and properties of ITC as shown so far are similar to those of the sum-of-squared-error criterion. Those will help us to understand ITC.

3.1. Generative model

In the background of information-theoretic clustering (ITC), there also exists the *bag-of-words* assumption [20] that disregards the order of words in a document. (Since ITC is not limited for document clustering, "word" is just an example of feature.) It means that features in data are conditionally independent and identically distributed, where the condition is a given probability distribution for an input vector. Based on this assumption, we describe a generative probabilistic model related to ITC and make clear the relationship between the model and the objective function (criterion) JS_W.

Let an input vector $\boldsymbol{x} = \{x_1,\ldots,x_m,\ldots,x_M\}$ present a set of the number of observations of mth feature. Suppose that there are clusters $C^k(k = 1,\ldots,K)$ which generates t^i features for a data (= input vector) with the probability distribution $\overline{P}^k = \{\overline{p}_1^k,\ldots,\overline{p}_M^k\}$, and conditional probability of a set of the observation about features in an input vector $\boldsymbol{x}^i$ is expressed by multinomial distribution

$$p(\boldsymbol{x}^i|\boldsymbol{x}^i{\in}C^k) = A^i \prod_{m=1}^{M} (\overline{p}_m^k)^{x_m^i}, \; A^i = \frac{t^i!}{x_1^i! \cdot x_2^i! \cdots x_M^i!}, \; t^i = \sum_{m=1}^{M} |x_m^i|, \tag{19}$$

where A^i is the number of combination of the observation. Assuming independence of each generation, joint probability function for the input vectors $\mathcal{X} = \{\boldsymbol{x}^1,\ldots,\boldsymbol{x}^N\}$ becomes

$$p(\mathcal{X}|\mathcal{C}) = \prod_{k=1}^{K} \prod_{\boldsymbol{x}^i \in C^k} A^i \prod_{m=1}^{M} (\overline{p}_m^k)^{x_m^i}, \tag{20}$$

where $\mathcal{C}$ indicates cluster information that specifies which input vector $\boldsymbol{x}^i$ belongs to cluster C^k. Taking the logarithm of Eq. (20) yields

$$\ln p(\mathcal{X}|\mathcal{C}) = \sum_{i=1}^{N} \log A^i + \sum_{k=1}^{K} \sum_{\boldsymbol{x}^i \in C^k} \sum_{m=1}^{M} \boldsymbol{x}_m^i \log \overline{p}_m^k \tag{21}$$

$$= \sum_{i=1}^{N} \log A^i + \sum_{k=1}^{K} \sum_{P^i \in C^k} \sum_{m=1}^{M} t^i \cdot p_m^i \log \overline{p}_m^k. \tag{22}$$

This is *a generative probabilistic model* related to ITC. If we assume that t^i takes constant value t for all input vectors, maximization of the probability (22) as well as minimization of the objective function JS_W (15) come to the minimization of

$$\frac{1}{N} \sum_{k=1}^{K} \sum_{P^i \in C^k} \sum_{m=1}^{M} -p_m^i \log \overline{p}_m^k = \frac{1}{N} \sum_{k=1}^{K} N_k \sum_{m=1}^{M} -\overline{p}_m^k \log \overline{p}_m^k, \tag{23}$$

for given input distribution $\mathcal{P}$. Here, the relationship $\sum_{P^i \in C^k} p_m^i = N_k \overline{p}_m^k$ is used. Since t^i may not be constant value t, the generative model (22) is not an equivalent model of ITC but the *related*

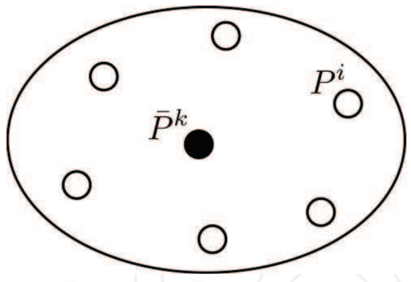

Figure 6. Input distributions and the mean in C^k.

model. This difference comes from the fact that the model treats each observation about features equally, while ITC treats each data (input vector) equally. Though the additional assumption $t^i = t$ is required, ITC works to find the most probable solution $\mathcal{C}$ in the generative probabilistic model. Furthermore, Eq. (23) is also based on the minimization of *entropy* in clusters as Eq. (23) shows. Entropy (specifically, Shannon Entropy) is the expected value of the information contained in each message that is an input distribution here. The smaller entropy becomes, and the more compactly a model can explain observations (input distributions). In this sense, the objective function JS_W (15) presents the goodness of the generative model. The relationship (including difference) between the probabilistic model and the objective function JS_W is meaningful to improve the model and the objective function in future.

Choice of appropriate model for data is important, when analyzing them. For example, large set of text documents contain many kinds of words and are presented as high-dimensional vectors. Taking extreme diversity of documents' topics into account, feature vectors of documents are distributed almost uniformly in the vector space. As known by **"the curse of dimensionality"** [10], most of the volume of a sphere in high-dimensional space is concentrated near the surface, and it becomes not appropriate to choose the model based on Gaussian distribution which concentrates values around the mean. In contrast, ITC on the basis of the multinomial distribution is a reasonable and useful tool to analyze such a high-dimensional count data, because the generative model of ITC is consistent with them.

We introduce **weight distribution** $\mathcal{Q}^k(k = 1,\ldots,K)(\mathcal{Q})$ that represent cluster C^k and that involves mean distribution $\overline{P}^k$ and prototype distribution of cluster C^k in a manner similar to that of the sum-of-squared-error (SSE) criterion (see Section2.2.1). **Figure 7** shows relationships between parameters in generative models. Parameters are generated or estimated by other parameters to maximize probability of generative model. For example, clustering is the task to find the most probable clusters $\mathcal{C}$ for given input vectors $\mathcal{X}$ or input distributions $\mathcal{P}$. In **Figure 7b**, constructing a classifier is the task to find $\mathcal{Q}$ for given $\mathcal{P}$ and $\mathcal{C}$ (classes in this context) in training process. Then, it estimates $\mathcal{C}$ for unknown $\mathcal{P}$ using the trained $\mathcal{Q}$. The classifier using multinominal distribution is known as multinominal *Naive Bayes classifier* [21]. As it shows, ITC and Naive Bayes classifier have a close relationship [18].

3.2. Algorithms

There exists difficulty (Appendix A) in algorithms for ITC. We show a novel idea to overcome it.

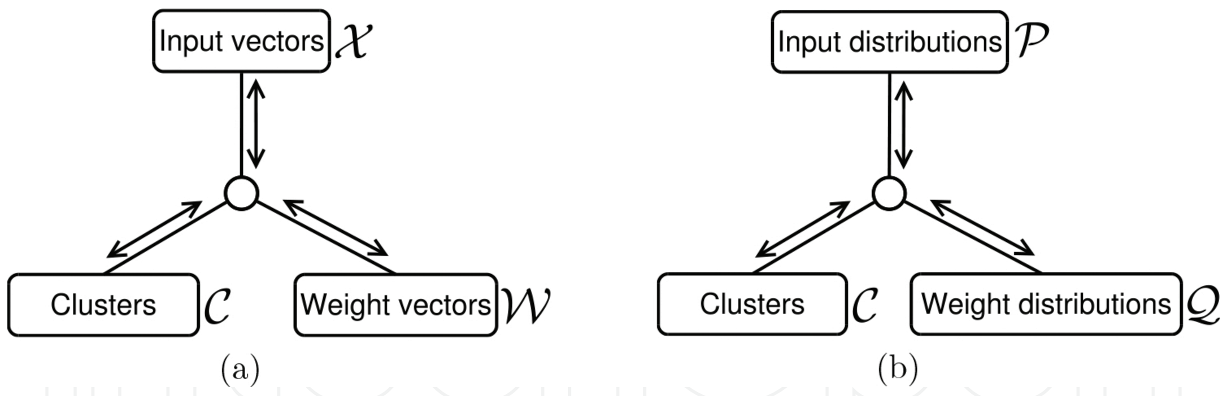
Input vectors $\mathcal{X}$
Clusters $\mathcal{C}$
Weight vectors $\mathcal{W}$
(a)
Input distributions $\mathcal{P}$
Clusters $\mathcal{C}$
Weight distributions $\mathcal{Q}$
(b)

Figure 7. Relationships between model parameters. (a) Clustering based on SSE criterion. (b) Information-theoretic clustering

3.2.1. Competitive learning

When competitive learning decides a winner for an input distribution P, it easily faces the difficulty of calculating KL divergence from P to weight distributions Q (see Appendix A). To overcome this difficulty, we present the idea to change an order of steps in competitive learning (CL). As shown in **Figure 8b**, CL updates all weights (= weight distributions) before deciding winner by

$$Q^k \leftarrow (1-\gamma)Q^k + \gamma P, \tag{24}$$

where γ is a learning rate. Since updated weight distributions $Q^k(k = 1,\ldots,K)$ include all words (features) of input distribution P, it is possible to calculate KL divergence $D_{\mathrm{KL}}(P\|Q^k)$ for all k. In following steps, CL decide a winner Q^c from $\mathcal{Q}$ by

$$c = \arg\min_k \ D_{\mathrm{KL}}(P\|Q^k) \ \ \text{(If there are several candidates, choose the smallest k)}, \tag{25}$$

and activate winner's update and discard others. These steps satisfy the CL's requirement that it partially reduces value of objective function JS_W in steepest direction with the given learning rate γ. Here, neither approximation nor distortion is added to the criterion of ITC. Note that updates of weight distributions Q^k before activation are provisional (see **Figure 8b**).

Related work that avoids the difficulty in calculating KL divergence presented **skew divergence** [22]. The skew divergence is defined as

$$s_\alpha(P,Q) = D_{\mathrm{KL}}(P\|\alpha Q + (1-\alpha)P), \tag{26}$$

where $\alpha(0 \le \alpha \le 1)$ is the mixture ratio of distributions. The skew divergence is exactly the KL divergence at $\alpha = 1$. When $\alpha = 1-\gamma$, Eq. (26) becomes similar to Eq. (24). Then, we can rewrite the steps in CL above using the skew divergence as

1. Select one input distribution P randomly from $\mathcal{P}$.
2. Decide a winner Q^c from $\mathcal{Q}$ by

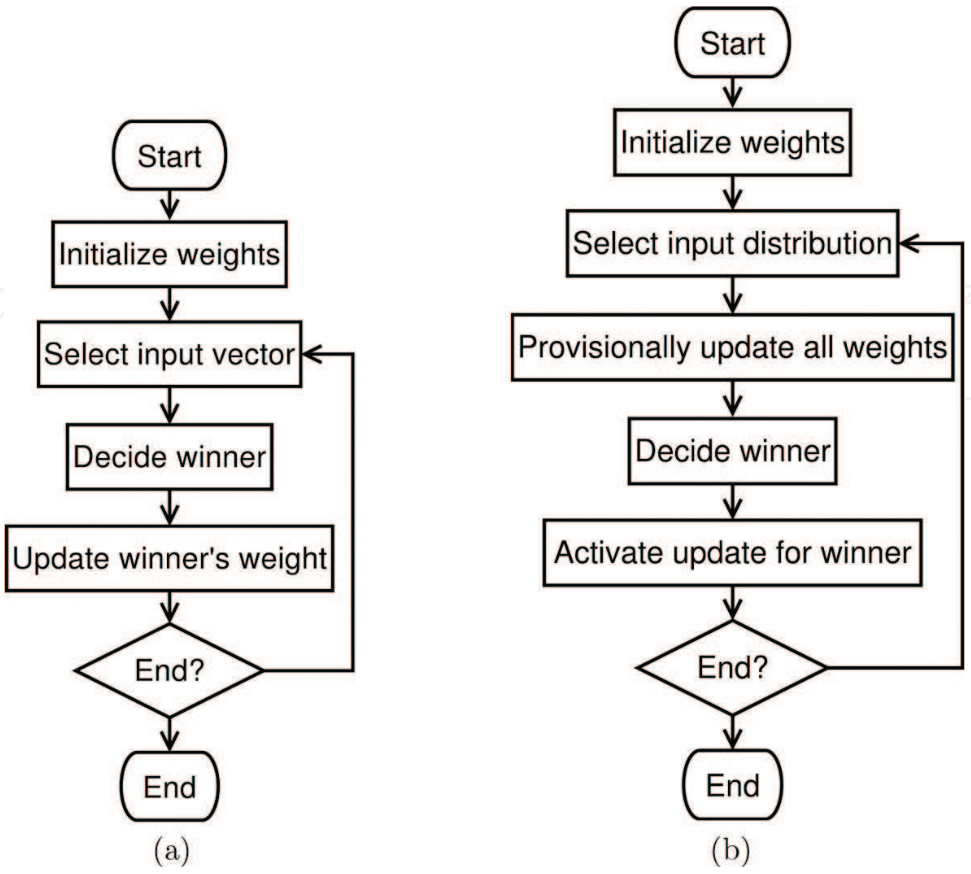

Figure 8. Flow charts of competitive learning. (a) Competitive learning for SSE. (b) Competitive learning for ITC

$$c = \arg\min_k \; s_\alpha(P,Q^k) \;\; \text{(If there are several candidates, choose the smallest k)}, \tag{27}$$

3. Update the winner's weight distribution Q^c as

$$Q^c \leftarrow (1-\gamma)Q^c + \gamma P, \tag{28}$$

where γ is a learning rate and equal to $1-\alpha$ (α is the mixture ratio for s_α) usually.

Hence, we call this novel algorithm for ITC as **“competitive learning using skew divergence”** (sdCL). In addition, splitting rule in competitive learning [12] can also be applied to this algorithm.

3.2.2. k-means type algorithm

Dhillon et al. [8] proposed *information-theoretic divisive algorithm* which is k-means type algorithm with divisive mechanism and uses KL divergence.[1] However, it still remains the

[1]The algorithm was proposed for feature/word clustering and applied to text classification. Since the algorithm uses document class (labeled data), it cannot be applied to general clustering problem.

difficulty to use KL divergence directly. In such a situation, we propose to use the skew divergence instead of KL divergence in k-means type algorithm as

1. Initialize clusters $\mathcal{C}$ of input distribution $\mathcal{P}$ randomly.
2. Update weight distributions $\mathcal{Q}$ by

$$Q^k = \frac{1}{N_k} \sum_{P^i \in C^k} P^i. \tag{29}$$

3. Update each cluster c of an input distribution P^i by

$$c = \arg\min_k \; s_\alpha(P^i, Q^k) \;\; \text{(If there are several candidates, choose the smallest k)}, \tag{30}$$

   where mixture ratio $\alpha(0 \leq \alpha \leq 1)$ for skew divergence s_α is 0.99 for example.

4. Repeat 2 and 3 until change ratio of objective function JS_W is less than small value (e.g., 10^{-8}).

The algorithm itself works well to obtain valuable clustering results. Further, if α is close to 1, skew divergence s_α becomes a good approximation of KL divergence. Therefore, restart of learning after termination with α closer to 1, such as $0.999, 0.9999, \ldots$, may lead to better clustering result.

3.2.3. Other algorithms

Slonim and Tishby [23] proposed an agglomerative hierarchical clustering algorithm, which is a hard clustering version of *Information Bottleneck algorithm* of Tishby et al. [24]. It is similar to the algorithm of Baker and McCallum [18] and merges just two clusters at every step based on the JS divergence of their distributions. A merit of the agglomerative algorithms is not affected by the difficulty of calculating KL divergence, because it just uses JS divergence. However, a merge of clusters at each step optimizes a local criterion but not a global criterion, as Dhillon et al. [8] pointed out. Therefore, clustering results may not be as good as results obtained by nonhierarchical algorithms (e.g., k-means and competitive learning) in the sense of optimizing the objective function of ITC. Additionally, hierarchical algorithms are computationally expensive, when the number of inputs is large.

Note that a lot of studies [8, 18, 23] aimed at improving accuracy of text classification using feature/word clustering based on ITC or distributional clustering. If a clustering is just a step to final goal, feature clustering is meaningful. However, features which characterize clusters should not be merged, when we aim to find clusters (topics) from a set of documents. Actually, finding topics using clustering is the aim of this chapter.

4. Evaluation of clustering

Since clustering results depend on methods (criteria and algorithms), appropriate selection of them is important. So far, we introduced two criteria for clustering. These are called **internal**

criteria that depend on their own models and not enough for evaluation. If criterion for clustering is common, we can compare clustering results by objective function of the criterion. Under a certain model that is an assumption in other word, a more probable result can be regarded as a better result. However, it is not guaranteed that the model or the assumption is reasonable at all times. Moreover, good clustering results under a certain criterion can be bad results under different criteria. A view from outside is required.

This section introduces **external criteria** that are **Purity**, **Rand index (RI)**, and **Normalized mutual information (NMI)** [25] to evaluate clustering quality and to find better clustering methods. These criteria compare clusters with a set of classes, which are produced on the basis of human judges. Here, each input data belong to one of class $A^j (j = 1,\ldots,J)$ and one of cluster $C^k (k = 1,\ldots,K)$. Let $T(C^k,A^j)$ be the number of data that belongs to both C^k and A^j.

Purity is measured by counting the number of input data from the most frequent class in each cluster. Purity can be computed as

$$\text{purity} = \frac{1}{N}\sum_{k=1}^{K} \max_{j} T(C^k,A^j), \tag{31}$$

where N is the total number of input data. Purity is close to 1, when each cluster has one dominant class.

Rand index (RI) checks all of the $N(N-1)/2$ pairs of input data and is defined by

$$\text{RI} = \frac{\mathrm{a}+\mathrm{b}}{\mathrm{a}+\mathrm{b}+\mathrm{c}+\mathrm{d}}, \tag{32}$$

where a, b, c, and d are the number of pairs in following conditions:

- "a," where the cluster number (suffix) is the same and the class number is the same
- "b," where the cluster numbers are different and the class numbers are different
- "c," where the cluster number is the same and the class numbers are different
- "d," where the cluster numbers are different and the class number is the same

The Rand index (RI) measures the percentage of agreements a+b in clusters and classes.

Normalized mutual information (NMI) is defined as

$$\text{NMI} = \frac{I(C;A)}{\big(H(C)+H(A)\big)/2}, \tag{33}$$

where, $I(C;A)$ is mutual information and $H()$ is entropy and

$$\begin{aligned} I(C;A) &= \sum_{k=1}^{K}\sum_{j=1}^{J} P(C^k,A^j)\log\frac{P(C^k,A^j)}{P(C^k)P(A^j)} \\ &= \sum_{k=1}^{K}\sum_{j=1}^{J}\frac{T(C^k,A^j)}{N}\log\frac{T(C^k,A^j)N}{T(C^k)T(A^j)}, \end{aligned} \tag{34}$$

$$H(C) = \sum_{k=1}^{K} -P(C^k)\log P(C^k) = \sum_{k=1}^{K} -\frac{T(C^k)}{N}\log\frac{T(C^k)}{N}, \tag{35}$$

$$H(A) = \sum_{j=1}^{J} -P(A^j)\log P(A^j) = \sum_{j=1}^{J} -\frac{T(A^j)}{N}\log\frac{T(A^j)}{N}, \tag{36}$$

where $P(C^k)$, $P(A^j)$, and $P(C^k,A^j)$ are the probability of data being in cluster C^k, class A^j, and in the intersection of C^k and A^j, respectively. Mutual information $I(C;A)$ measures the mutual dependence between clusters C and classes A. It quantifies the amount of information obtained for classes through knowing about clusters. Hence, high NMI shows some kind of goodness about clustering in information theory.

5. Experiments

This section provides experimental results that show the effectiveness and usefulness of ITC and the proposed algorithm (sdCL: competitive learning using skew divergence). Experiments consist of two parts, experiment1 and experiment2.

In experiment1, we applied sdCL to the same data sets as used in the paper of Wang et al. [26] and compared performance of sdCL with other clustering algorithms evaluated in it. The algorithms that the paper [26] evaluated are as follows.

- **k-means (KM)**. The weights $\mathcal{W}$ are initialized by randomly [26].
- **Normalized cut (NC)** [27].
- **Maximum margin clustering (MMC)** [7].
- **Generalized maximum margin clustering (GMMC)** [28].
- **Iterative support vector regression (IterSVR)** [29].
- **Cutting plane maximum margin clustering (CPMMC)** [26], **CPM3C** [26].

As shown above, *maximum margin clustering* (MMC) [7] and related works are much focused. These works extend the idea of support vector machine (SVM) [30] to the unsupervised scenario. The experimental results obtained by the MMC technique are often better than conventional clustering methods. Among those, CPMMC and CPM3C (Cutting plane multiclass maximum margin clustering) [26] are known as successful methods. Experimental results will show that the proposed algorithm sdCL outperforms CPM3C in text data clustering.

In experiment2, we focus on text data clustering and compare performance of algorithms, sdCL, sdCLS (sdCL with splitting rule, see Sections 2.2.2 and 3.2.1), and spherical competitive learning (spCL). We also provide the evidence to support the effectiveness of ITC by detailed analysis of clustering results.

spCL is an algorithm for spherical clustering like the spherical k-means algorithm [5] that was proposed for clustering high-dimensional and sparse data, such as text data. The objective function to be maximized for the spherical clustering is cosine similarity between input vectors and the mean vector of a cluster to which they belong. To implement spCL, we turn input and weight vectors ($\boldsymbol{x}$, $\boldsymbol{w}$) into a unit vector and decide winner $\boldsymbol{w}^c$ by

$$c = \arg\max_k \cos(\boldsymbol{x},\boldsymbol{w}^k) \text{ (If there are several candidates, choose the smallest k)}, \tag{37}$$

and update the winner's weight $\boldsymbol{w}^c$ as

$$\boldsymbol{w}^c \leftarrow \frac{(1-\gamma)\boldsymbol{w}^c + \gamma\boldsymbol{x}}{\|(1-\gamma)\boldsymbol{w}^c + \gamma\boldsymbol{x}\|}. \tag{38}$$

For all competitive learning algorithms, the learning rate $\gamma = 0.01$, the number of maximum repetitions for updating weights $N_r = 1,000,000$ (termination condition), and the threshold of times for splitting rule $\theta = 1000$ are used. After competitive learning (sdCL, sdCLS, or spCL) is terminated, we apply k-means type algorithm to remove fluctuation as a post-processing. Specifically, sdKM (the k-means type algorithm using skew divergence shown in Section 3.2.2) with $\alpha = 0.999, 0.9999, 0.99999$ is applied consecutively after sdCL and sdCLS. In each learning procedure including post-processing, an operation is iterated 50 times with different initial random seeds for a given set of parameters.

5.1. Data sets

We mainly use the same data sets as used in the paper of Wang et al. [26]. When applying algorithms for ITC, we use probability distributions $P^i(i = 1,\ldots N)$ ($\mathcal{P}$) derived from original data.

1. **UCI data**. From the UCI repository,[2] we use ionosphere, digits, letter, and satellite under the same setting of the paper [26]. The digits data (8×8 matrix) are generated from bitmaps of handwritten digits. Pairs (3 vs. 8, 1 vs. 7, 2 vs. 7, and 8 vs. 9) are focused due to the difficulty of differentiating. For the letter and satellite data sets, their first two classes are used. Since the ionosphere data contain minus values and cannot be transformed to probability distributions, we do not apply ITC to them.

[2] http://archive.ics.uci.edu/ml/[Accessed: 2016-10-25].

2. **Text data**. Four text data sets: 20Newsgroups (http://qwone.com/~jason/20Newsgroups/ [Accessed: 2016-10-25]), WebKB,[3] Cora [31], and RCV1 (Reuters Corpus Volume 1) [32] are used. In experiment1, we follow the setting of the paper [26]. For 20Newsgroups data set, topic "rec" which contains four topics {autos, motorcycles, baseball, hockey} is used. From the four topics, two sets of two-class data sets {Text-1: autos vs motorcycles, Text-2: baseball vs hockey} are extracted. From WebKB data sets, the four Universities data set (Cornell, Texas, Washington, and Wisconsin University), which has seven classes (student, faculty, staff, department, course, project, and other), are used. Note that topic of the "other" class is ambiguous and may contain various topics (e.g., faculty), because it is a collection of pages that were not deemed the "main page" representing an instance of the other six classes, as pointed out in the web page of the data set. Cora data set (Cora research paper classification) [31] is a set of information of research papers classified into a topic hierarchy. From this data set, papers in subfield {data structure (DS), hardware and architecture (HA), machine learning (ML), operating system (OS), programming language (PL)} are used. We select papers that contain title and abstract. RCV1 data set contains more than 800 thousands documents to which topic category is assigned. The documents with the highest four topic codes (CCAT, ECAT, GCAT, and MCAT) in the topic codes hierarchy in the training set. Multi-labeled instances are removed.

   In experiment2, we use all of 20Newsgroups and RCV1 data sets. For RCV1 data set, we obtain 53 classes (categories) by mapping the data set to the second level of RCV1 topic hierarchy and remove multi-labeled instances. For WebKB data set, we remove "other" class due to ambiguity, use the other six classes, and do not use information of universities.

   For all text data, we remove *stop words* using *stop list* [32] and empty data, if they are not removed. In experiment1, we follow the setting of the paper [26], but properties of data sets are slightly different (see **Table 3**). For Cora data sets, the differences of data sizes are large. However, they must keep the same (or close at least) characteristics (e.g., distributions of words and topics), because they are extracted from the same source.

3. **Digits data**. USPS (16×16) and MNIST (28×28) are data sets of handwritten digits image.[4] For USPS data set, 1, 2, 3, and 4 digit images are used. For MNIST and digits data from UCI repository, all 45 pairs of digits 0–9 are used in two-class problems.

   The properties of those data sets are listed in **Table 3**.

5.2. Results of experiment1

The clustering results are shown in **Tables 4–7**, where values (except for sdCL) are the same in the paper of Wang et al. [26] (*accuracy* in that paper is equivalent to *purity* from its definition). In two-class problems, CPMMC outperforms other algorithms about purity and Rand Index (RI) in most cases. The proposed algorithm sdCL shows stable performances except for

[3]http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/[Accessed: 2016-10-25].
[4]http://www.kernel-machines.org/data [Accessed: 2016-10-25].

Data	Size (N)	Feature (M)	Class (K)
Ionosphere	351	34	2
Letter	1555	16	2
Digits	1555	64	2
Satellite	2236	36	2
Text-1	1981	16,259	2
Text-2	1987	15,955	2
20Newsgroups-4	3967	24,506	4
20Newsgroups-20	18,772	60,698	20
Cora-DS	2397	5745	9
Cora-HA	913	3340	7
Cora-ML	3569	6809	7
Cora-OS	2084	5029	4
Cora-PL	3026	6069	9
WebKB-Cornell	835	5574	7
WebKB-Texas	808	4482	7
WebKB-Washington	1191	7779	7
WebKB-Wisconsin	1218	8270	7
WebKB6	4219	14,142	6
Reuters-RCV1-4	19,806	44,214	4
Reuters-RCV1-53	534,135	216,704	53
MNIST	70,000	784	2
USPS	3046	256	4

Table 3. Properties of data sets.

ionosphere to which sdCL cannot be applied. In multiclass problems, sdCL for text data (Cora, 20Newsgroups-4, and Reuters-RCV1-4) outperforms other algorithms. The results show that ITC and the proposed algorithm sdCL are effective for text data sets. Note that CPM3C shows the better results than sdCL for WebKB data. However, topic of the "other" class in WebKB is ambiguous (see Section5.1). The occupation ratio of them is large {0.710, 0.689, 0.777, 0.739} and almost same as the values of purity in CPM3C and sdCL. It means that these algorithms failed to find meaningful clusters in purity. Therefore, WebKB data are not appropriate to use for evaluation without removing "other" class.

5.3. Results of experiment2

In experiment2, we focus on text data clustering. **Table 8** shows that the proposed algorithms for ITC (sdCL and sdCLS) outperform spCL in purity, RI, and NMI. Considering that spCL is an algorithm for spherical clustering [5] which was proposed to analyze high-dimensional

Data	KM	NC	MMC	GMMC	IterSVR	CPMMC	sdCL
Iono	0.5428	0.7500	**0.7875**	0.7650	0.7770	0.7548	–
Letter	0.8206	0.7680	–	–	0.9280	**0.9502**	0.9267
Satellite	0.9593	0.9579	–	–	0.9682	**0.9879**	0.9506
Text-1	0.5053	0.9379	–	–	**0.9702**	0.9500	0.9306
Text-2	0.5038	0.9135	–	–	0.9399	**0.9721**	0.9591
Dig 3–8	0.9468	0.6500	0.9000	0.9440	0.9664	**0.9688**	0.9440
Dig 1–7	0.9445	0.5500	0.6875	0.9780	0.9945	**1.000**	**1.000**
Dig 2–7	0.9691	0.6600	0.9875	0.9950	**1.000**	**1.000**	0.9891
Dig 8–9	0.9068	0.5200	0.9625	0.8400	0.9633	**0.9812**	0.8910
UCI-dig	0.9638	0.9757	–	–	0.9818	**0.9940**	0.9516
MNIST	0.8921	0.8992	–	–	0.9241	**0.9621**	0.8812

Bold fonts indicate the maximum purities for a give data set.

Table 4. Purity comparisons for two-class problems.

Data	KM	NC	MMC	GMMC	IterSVR	CPMMC	sdCL
Iono	0.56	0.63	**0.67**	0.64	0.56	0.65	–
Letter	0.71	0.64	–	–	0.87	**0.92**	0.86
Satellite	0.92	0.92	–	–	0.94	**0.97**	0.91
Text-1	0.50	0.88	–	–	**0.94**	**0.94**	0.87
Text-2	0.50	0.84	–	–	0.89	**0.93**	0.92
Dig 3–8	0.90	0.55	0.82	0.90	0.94	**0.95**	0.89
Dig 1–7	0.99	0.50	0.57	0.96	0.99	**1.0**	**1.0**
Dig 2–7	0.94	0.55	0.98	0.99	**1.0**	**1.0**	0.98
Dig 8–9	0.84	0.50	0.93	0.73	0.93	**0.97**	0.81
UCI-dig	0.93	0.96	–	–	0.97	**0.99**	0.93
MNIST	0.81	0.82	–	–	0.86	**0.94**	0.83

Bold fonts indicate the maximum rand indices for a give data set.

Table 5. Rand Index (RI) comparisons for two-class problems.

data such as text documents, the criterion of information-theoretic clustering is worth to use for this purpose.

Table 8 also shows that sdCLS (sdCL with splitting rule, see Sections 2.2.2 and 3.2.1) is slightly better than sdCL in some cases. As far as **Figure 9** (left, right) shows, values of JS divergence for sdCLS are smaller ("better" in ITC) than sdCL, and sdCLS outperforms sdCL in purity on average. Nevertheless, an advantage of sdCLS against sdCL is not so obvious in this experiment.

Data	KM	NC	MMC	CPM3C	sdCL
UCI-digits 0689	0.4223	0.9313	0.9483	**0.9674**	0.9394
UCI-digits 1279	0.4042	0.9011	0.9191	**0.9452**	0.8300
USPS	0.9215	0.9011	0.9191	0.9452	**0.9515**
Cora-DS	0.2824	0.3688	–	0.4415	**0.5057**
Cora-HA	0.3402	0.4200	–	0.5980	**0.6145**
Cora-ML	0.2708	0.3103	–	0.4549	**0.5974**
Cora-OS	0.2387	0.2303	–	0.5916	**0.6686**
Cora-PL	0.3380	0.3397	–	0.4721	**0.4729**
WebKB-Cornell	0.5571	0.6143	–	**0.7205**	0.7192
WebKB-Texas	0.4505	0.3538	–	**0.6910**	0.6895
WebKB-Washington	0.5352	0.3285	–	**0.7817**	0.7767
WebKB-Wisconsin	0.4953	0.3331	–	**0.7425**	0.7397
20Newsgroups-4	0.3527	0.4189	–	0.7134	**0.9360**
Reuters-RCV1-4	0.2705	–	–	0.6235	**0.8064**

Bold fonts indicate the maximum purities for a give data set.

Table 6. Purity comparisons for multiclass problems.

Data	KM	NC	MMC	CPM3C	sdCL
UCI-digits 0689	0.696	0.939	0.941	**0.974**	0.945
UCI-digits 1279	0.4042	0.9011	0.9191	**0.945**	0.868
USPS	0.932	0.938	–	0.950	**0.958**
Cora-DS	0.589	0.744	–	0.746	**0.823**
Cora-HA	0.385	0.659	–	0.695	**0.767**
Cora-ML	0.514	0.720	–	0.761	**0.802**
Cora-OS	0.518	0.522	–	0.730	**0.735**
Cora-PL	0.643	0.675	–	0.712	**0.819**
WebKB-Cornell	0.603	0.602	–	**0.724**	0.483
WebKB-Texas	0.604	0.602	–	**0.712**	0.495
WebKB-Washington	0.616	0.581	–	**0.752**	0.426
WebKB-Wisconsin	0.581	0.509	–	**0.761**	0.464
20Newsgroups-4	0.581	0.496	–	0.780	**0.940**
Reuters-RCV1-4	0.471	–	–	0.703	**0.800**

Bold fonts indicate the maximum rand indices for a give data set.

Table 7. Rand Index (RI) comparisons for multiclass problems.

	sdCL			sdCLS			spCL		
Data	**Purity**	**RI**	**NMI**	**Purity**	**RI**	**NMI**	**Purity**	**RI**	**NMI**
Cora-DS	0.506	0.823	0.322	0.514	0.826	0.327	0.361	0.796	0.162
Cora-HA	0.614	0.767	0.363	0.618	0.767	0.360	0.518	0.728	0.251
Cora-ML	0.597	0.802	0.384	0.602	0.801	0.385	0.431	0.757	0.181
Cora-OS	0.669	0.735	0.313	0.685	0.741	0.325	0.582	0.673	0.187
Cora-PL	0.473	0.819	0.275	0.484	0.820	0.274	0.414	0.802	0.175
WebKB6	0.575	0.712	0.272	0.575	0.712	0.272	0.519	0.702	0.181
20News-20	0.690	0.954	0.653	0.697	0.955	0.656	0.359	0.905	0.324
RCV1-53	0.731	0.910	0.586	0.738	0.906	0.580	0.568	0.895	0.384

Table 8. Comparison for text data sets.

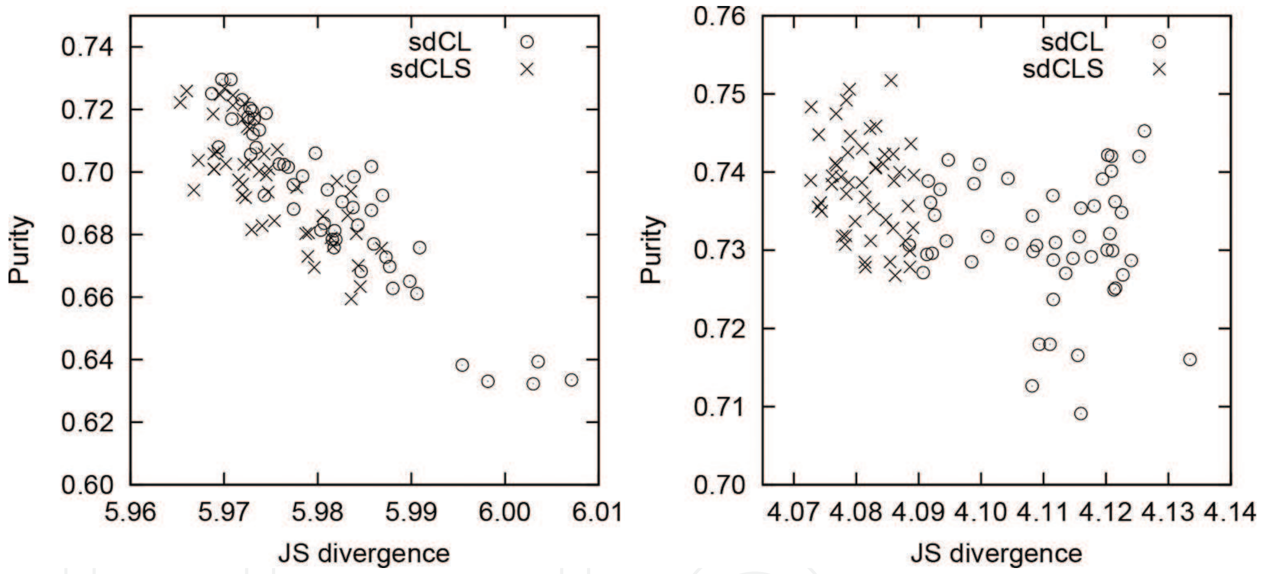

Figure 9. Purity versus JS divergence for 20Newsgroups (left) and Reuters-RCV1 (right) data sets.

Note that clustering result by dCL (competitive learning using **KL divergence**) is shown below. Since the values of NMI clearly illustrate that dCL converged to unrelated solutions to the classes, use of **skew divergence** is an effective technique to overcome this problem.

Data	**Purity**	**RI**	**NMI**
WebKB6	0.363	0.667	0.00629
20News-20	0.071	0.905	0.00596

alt.atheism	god writes people article atheism religion time evidence
comp.graphics	image graphics jpeg file bit images software data files ftp
comp.os.ms-windows.misc	windows file dos writes article files ms os problem win
comp.sys.ibm.pc.hardware	drive scsi card mb ide system controller bus pc writes
comp.sys.mac.hardware	mac apple writes drive system problem article mb monitor mhz
comp.windows.x	window file server windows program dos motif sun display widget
misc.forsale	sale shipping offer mail price drive condition dos st email
rec.autos	car writes article cars good engine apr ve people time
rec.motorcycles	writes bike article dod ca apr ve ride good time
rec.sport.baseball	writes year article game team baseball good games time hit
rec.sport.hockey	game team hockey writes play ca games article season year
sci.crypt	key encryption government chip writes clipper people article keys system
sci.electronics	writes article power good ve work ground time circuit ca
sci.med	writes article people medical health disease time cancer patients
sci.space	space writes nasa article earth launch orbit shuttle time system
soc.religion.christian	god people jesus church christ writes christian christians bible time
talk.politics.guns	gun people writes article guns fbi government fire time weapons
talk.politics.mideast	people israel armenian writes turkish jews article armenians israeli jewish
talk.politics.misc	people writes article president government mr stephanopoulos make time
talk.religion.misc	god writes people jesus article bible christian good christ life

Table 9. Frequent words in classes of 20Newsgroups data set.

In followings, we examine inside of clustering results obtained by ITC to make clear whether ITC helps us to find meaningful clusters and candidates of classes for classification. **Tables 9** and **10** show frequent words in classes and clusters obtained by sdCL of 20Newsgroups data set, respectively. The order of clusters is arranged so that clusters are made to correspond to classes. **Table 11** is the cross table between clusters and classes. As shown in **Table 10**, the frequent words in some clusters remind us characteristics of them to distinguish from others. For example, the words in cluster 2: "image graphics jpeg," cluster 6: "sale offer shipping," and cluster 11: "key encryption chip" remind classes (comp.graphics), (misc.forsale), and (sci.crypt), respectively. We also imagine characteristics of clusters from the words in 7, 8, 9, 10, 13, 14, and 16th clusters. These clusters have documents of one dominant class and can be regarded as candidates of classes. However, there are some exceptions. The 1st and 15th clusters have the same word "god," while classes of (alt.atheism), (soc.religion.christian), and (talk.religion.misc) have also the same word "god." The cluster 1 and class (alt.atheism) have common words "religion evidence," and the cluster 1 has many documents of the dominant class (alt.atheism). The cluster 15 and the class (soc.religion.christian) have common words "jesus bible christ church," and the cluster 15 has many documents of the dominant class (soc.religion.christian). On the other hand, there is no cluster which has

1	writes god article people religion evidence science moral objective time
2	image graphics jpeg file images ftp data bit files format
3	windows dos file system writes os files article program software
4	drive scsi mb card writes system mac article bit apple
5	window file server program motif sun widget set display output
6	sale offer shipping mail price condition st good email interested
7	car writes article cars engine good ve apr time oil
8	writes article bike dod apr ca ride ve good time
9	writes year article game team baseball good games time hit
10	game team hockey writes play ca games article year season
11	key encryption government chip writes clipper people article keys system
12	db writes article power good ground ca time ve circuit
13	writes article medical people disease health cancer patients time msg
14	space writes nasa article earth launch orbit time shuttle system
15	god jesus people bible writes christ christian church christians article
16	people writes gun article government fbi fire guns koresh batf
17	israel israeli writes jews article people arab jewish arabs state
18	people armenian turkish armenians president armenia war mr turks turkey
19	writes people article government cramer apr state make health optilink
20	mail list address email send information ca internet article writes

Table 10. Frequent words in clusters of 20Newsgroups data set.

documents of (talk.religion.misc) as dominant, and the documents of (talk.religion.misc) are mostly shared by the clusters 1 and 15. Though there is a mismatch between clusters and classes, the clustering result is also acceptable, because words in the class (talk.religion.misc) are resemble those in the class (soc.religion.christian). We can also find that cluster 4 has many documents of the two classes (comp.sys.ibm.pc.hardware) and (comp.sys.mac.hardware). From **Table 9**, those classes have similar words except for "mac" and "apple." Thus, ITC missed to detect the difference of the classes, but found the cluster with common feature of them. In this sense, the clustering result is meaningful and useful. Another example is that documents in class (talk.politics.mideast) are divided into clusters 17 and 18. It means that ITC found two topics from one class and frequency words in the clusters seem to be reasonable (see 17th and 18th clusters in **Table 10**). The characteristic of cluster 20 that has words "mail list address email send" is different from all classes as well as other clusters, but the cluster 20 has some documents in all classes (see **Table 11**). This cluster may discover that all newsgroups include documents with such words. In summary, ITC helps us to find meaningful clusters, even when clusters obtained by ITC sometimes seem not to be the same as expected classes. The detailed analysis of the clustering results above could be the evidence to support the effectiveness and usefulness of ITC.

1	625	4	9	2	2	4	2	8	0	3	2	0	5	49	12	54	1	5	13	163
2	1	632	44	19	17	101	10	7	4	8	0	14	32	12	18	2	0	0	1	1
3	0	107	707	189	58	60	27	5	0	0	1	12	40	2	3	3	0	0	1	1
4	0	74	64	630	711	18	104	3	3	1	0	2	93	6	3	2	0	1	0	1
5	1	43	53	11	15	712	0	1	0	1	1	9	2	1	1	0	1	0	0	3
6	0	9	5	18	25	2	648	17	15	3	0	0	16	1	3	1	1	0	1	0
7	0	1	2	7	8	0	35	784	65	1	1	0	25	2	7	2	2	2	1	1
8	1	3	0	0	4	6	7	36	867	2	0	2	10	6	6	0	1	1	0	2
9	4	0	4	4	0	1	4	3	4	886	18	0	2	3	2	3	0	3	6	3
10	0	0	1	1	0	1	9	3	2	37	943	0	1	1	1	0	3	2	3	0
11	2	15	2	4	6	5	5	1	0	1	2	881	36	5	6	0	20	9	15	3
12	1	11	11	52	33	2	32	14	4	3	0	8	603	28	10	2	1	0	0	1
13	1	0	1	0	5	1	3	1	1	1	2	1	5	771	5	3	0	0	2	2
14	3	9	5	2	3	5	4	4	4	1	1	1	29	17	843	2	4	0	5	5
15	94	1	2	0	0	0	2	1	1	3	0	0	0	10	2	863	5	4	6	299
16	15	2	0	5	8	2	6	36	11	0	1	18	6	8	4	7	805	2	189	93
17	9	2	1	0	1	1	0	7	0	0	0	0	2	1	8	6	2	547	14	10
18	32	0	0	0	0	0	1	1	0	2	1	1	5	3	4	12	12	333	38	12
19	3	7	8	2	16	3	12	20	4	4	7	13	6	13	11	13	33	12	470	16
20	6	50	44	33	46	58	53	35	8	34	16	27	66	48	36	22	18	19	8	11

Table 11. The number of documents about each class in each cluster.

6. Conclusion

In this chapter, we introduced information-theoretic clustering (ITC) from both theoretical and experimental side. Theoretically, we have shown the criterion, generative model, and novel algorithms for ITC. Experimentally, we showed the effectiveness and usefulness of ITC for text analysis as an important example.

A Difficulty about KL divergence

Let P and Q be a distribution whose mth random variable p_m and q_m takes the mth element of a vector $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively. The Kullback-Leibler (KL) divergence to Q from P is defined to be

$$D_{\mathrm{KL}}(P\|Q) = p_m \log \frac{p_m}{q_m}. \tag{39}$$

In this definition, it is assumed that the support set of P is a subset of the support set of Q (If q_m is zero, p_m must be zero). For a given cluster C^k, there is no problem to calculate JS divergence of cluster C^k by Eq. (12), because the support set of any distribution $P^i(\in C^k)$ is the subset of the mean distribution $\overline{P}^k$. However, it is not guaranteed that KL divergence from $P^i(\in C^k)$ to $Q^t(t\neq k)$ (a weight distribution of other cluster C^t) is finite. This causes a serious problem to find similar weight distribution Q for an input distribution P. For example, lack of even one word (feature) in a distribution Q is enough not to be similar. Therefore, it is difficult to use k-means type algorithm,[5] which updates weights or clusters by batch processing, in ITC.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number 26330259.

Author details

Toshio Uchiyama

Address all correspondence to: uchiyama.toshio@do-johodai.ac.jp

Hokkaido Information University, Ebetsu-shi, Hokkaido, Japan

References

[1] A.K. Jain, M.N. Murty and P.J. Flynn, “Data clustering: a review,” ACM Computing Surveys (CSUR), vol. 31, no. 3, pp. 264–323, 1999.

[2] A.K. Jain, “Data clustering: 50 years beyond k-means,” Pattern Recognition Letters, vol. 31, no. 8, pp. 651–666, 2010.

[3] S. Lloyd, “Least squares quantization in pcm,” IEEE Transactions on Information Theory, vol. 28, no. 2, pp. 129–137, 1982.

[4] D.E. Rumelhart and D. Zipser, “Feature discovery by competitive learning,” Cognitive Science, vol. 9, pp. 75–112, 1985.

[5] I.S. Dhillon and D.S. Modha, “Concept decompositions for large sparse text data using clustering,” Machine Learning, vol. 42, no. 1–2, pp. 143–175, 2001.

[6] A.Y. Ng, M.I. Jordan, Y. Weiss, et al., “On spectral clustering: analysis and an algorithm,” Advances in Neural Information Processing Systems, vol. 2, pp. 849–856, 2002.

[5]k-means algorithm is known as algorithm for clustering based on the sum-of-squared-error criterion (see Section 2.2.1). Therefore we added word “type” for information-theoretic clustering.

[7] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans, "Maximum margin clustering," Advances in Neural Information Processing Systems, vol. 17, pp. 1537–1544, 2004.

[8] I.S. Dhillon, S. Mallela, and R. Kumar, "A divisive information theoretic feature clustering algorithm for text classification," The Journal of Machine Learning Research, vol. 3, pp. 1265–1287, 2003.

[9] R.O. Duda and P.E. Hart, Pattern classification and scene analysis. John Wiley & Sons, New York, 1973.

[10] C.M. Bishop, Pattern recognition and machine learning (Information Science and Statistics), 1st edn. 2006. corr. 2nd printing edn. Springer, New York, 2007.

[11] J. MacQueen, et al., "Some methods for classification and analysis of multivariate observations," Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol. 1, Oakland, CA, USA., pp. 281–297, 1967.

[12] T. Uchiyama and M.A. Arbib, "Color image segmentation using competitive learning," The IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 16, no. 12, pp. 1197–1206, 1994.

[13] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design," IEEE Transactions on Communications, vol. 28, no. 1, pp. 84–95, 1980.

[14] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms Society for Industrial and Applied Mathematics, pp. 1027–1035, 2007.

[15] S. Amari, "A theory of adaptive pattern classifiers," IEEE Transactions on Electronic Computers, vol. 16, no. 3, pp. 299–307, 1967.

[16] S.-I. Amari, "Natural gradient works efficiently in learning," Neural Computation, vol. 10, no. 2, pp. 251–276, 1998.

[17] F. Pereira, N. Tishby, and L. Lee, "Distributional clustering of english words," In: Proceedings of the 31st annual meeting on Association for Computational Linguistics Association for Computational Linguistics, pp. 183–190, 1993.

[18] L.D. Baker and A.K. McCallum, "Distributional clustering of words for text classification," In: Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval ACM, pp. 96–103, 1998.

[19] N. Slonim and N. Tishby, "Document clustering using word clusters via the information bottleneck method," In: Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval ACM, pp. 208–215, 2000.

[20] D.M. Blei, A.Y. Ng, and M.I. Jordan, "Latent dirichlet allocation," Journal of Machine Learning Research, vol. 3, pp. 993–1022, 2003.

[21] A. McCallum and K. Nigam, "A comparison of event models for naive Bayes text classification," In: AAAI-98 Workshop on Learning for Text Categorization, vol. 752, Citeseer, pp. 41–48, 1998.

[22] L. Lee, "Measures of distributional similarity," In: Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics Association for Computational Linguistics, pp. 25–32, 1999.

[23] N. Slonim and N. Tishby, "The power of word clusters for text classification," In: 23rd European Colloquium on Information Retrieval Research, 2001.

[24] N. Tishby, F.C. Pereira, and W. Bialek, "The information bottleneck method," In: The 37th annual allerton conference on communication, control, and computing, pp. 368–377, 1999.

[25] C.D. Manning, P. Raghavan, and H. Schütze, Introduction to information retrieval, vol. 1, Cambridge University Press, Cambridge, 2008.

[26] F. Wang, B. Zhao, and C. Zhang, "Linear time maximum margin clustering," IEEE Transactions on Neural Networks, vol. 21, no. 2, pp. 319–332, 2010.

[27] J. Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pp. 888–905, 2000.

[28] H. Valizadegan and R. Jin, "Generalized maximum margin clustering and unsupervised kernel learning," Advances in Neural Information Processing Systems, pp. 1417–1424, 2006.

[29] K. Zhang, I.W. Tsang, and J.T. Kwok, "Maximum margin clustering made practical," IEEE Transactions on Neural Networks, vol. 20, no. 4, pp. 583–596, 2009.

[30] B. Schölkopf and A.J. Smola, Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge MA, London, 2002.

[31] A.K. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of internet portals with machine learning," Information Retrieval, vol. 3, no. 2, pp. 127–163, 2000. https://people.cs.umass.edu/~mccallum/data.html

[32] D.D. Lewis, Y. Yang, T.G. Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," The Journal of Machine Learning Research, vol. 5, pp. 361–397, 2004.