

On Quantitative Evaluation of Clustering Systems

Ji He*, Ah-Hwee Tan[‡], Chew-Lim Tan*, Sam-Yuan Sung*

**School of Computing, National University of Singapore
3 Science Drive 2, Singapore 117543
E-mail: {heji,tancl,ssung}@comp.nus.edu.sg*

*‡Laboratories for Information Technology
21 Heng Mui Keng Terrace, Singapore 119613
E-mail: ahhwee@lit.org.sg*

Contents

1	Introduction	2
2	Clustering Process: A Brief Review	3
2.1	Pattern Representation, Feature Selection and Feature Extraction . . .	3
2.2	Pattern Proximity Measure	5
2.3	Clustering Algorithms	6
3	Evaluation of Clustering Quality	8
3.1	Evaluation Measures Based on Cluster Distribution	10
3.2	Evaluation Measures Based on Class Conformation	12
4	Clustering Methods	13
4.1	k-means	14
4.2	Self-Organizing Maps (SOM)	14
4.3	Adaptive Resonance Theory under Constraints (ART-C)	15

5 Experiments and Discussions	18
5.1 Statistical Validation of Comparative Observation	19
5.2 Identification of the Optimal Number of Clusters	19
5.3 Selection of Pattern Proximity Measure	21
5.4 Cross-Comparison of Clustering Methods	22
5.4.1 Data Preparation	22
5.4.2 Evaluation Paradigm	23
5.4.3 Results and Discussions	24
6 Conclusions	26
References	

1 Introduction

Clustering refers to the task of partitioning unlabelled data into meaningful groups (clusters). It is a useful approach in data mining processes for identifying hidden patterns and revealing underlying knowledge from large data collections. The application areas of clustering, to name a few, include image segmentation, information retrieval, document classification, associate rule mining, web usage tracking, and transaction analysis.

While a large number of clustering methods have been developed, clustering remains a challenging task as a clustering algorithm behaves differently depending on the chosen features of the data set and the parameter values of the algorithm [10]. Therefore, it is important to have some objective measures to evaluate the clustering quality in a quantitative manner. Given a clustering problem with no prior knowledge, a quantitative assessment of the clustering algorithm serves as an important reference for various tasks, such as discovering the distribution of a data set, identifying the clustering paradigm that is most suitable for a problem domain, and deciding the optimal parameters for a specific clustering method.

In the rest of this chapter, we review the process of clustering activities and discuss the factors that affect the output of a clustering system. We then describe two sets of quality measures for the evaluation of clustering algorithms. While the first set of measures evaluate clustering outputs in terms of their inherent data distribution, the second set evaluates output clusters in terms of how well they follow known class distribution of the

problem domain. We illustrate the application of these evaluation measures through a series of controlled experiments using three clustering algorithms.

2 Clustering Process: A Brief Review

Jain et al summarized a typical sequence of the clustering activities as the following three stages depicted in Figure 1 [13, 14]:

1. *pattern representation*, optionally including feature extraction and/or selection,
2. definition of a *pattern proximity measure* for the data domain, and
3. *clustering* or *grouping* of data points according to the chosen pattern representation and the proximity measure.

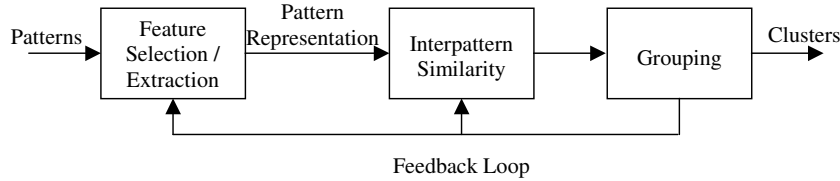


Figure 1: A typical sequencing of clustering activity.

Since the output of a clustering system is the result of the system’s interactive activities in each stage, various factors in each stage that affect the system’s activity in turn have impact on the clustering output. We extend our discussion in the following subsections.

2.1 Pattern Representation, Feature Selection and Feature Extraction

Pattern representation refers to the paradigm for observation and the abstraction of the learning problem, including the type, the number and the scale of the features, the number of the patterns, and the format of the feature representation. Feature selection is defined as the task of identifying a set of most representative subset of the natural features (or transformations of the natural features) to be used by the machine. Feature extraction, on

the other hand, refers to the paradigm for converting the observations of the natural features into a machine understandable format.

Pattern representation is considered as the basis of machine learning. Since human accessibility of the patterns is highly dependent on their representation format, an unsuitable pattern representation may result in a failure of producing meaningful clusters as a user desires. As show in Figure 2a, using a cartesian coordinate representation, a clustering method would have no problem in identify the five compact groups of data points. However, when the same representation is applied to the data set in Figure 2b, the four string-shape clusters would probably not be discovered as they are not easily separable, in terms of Euclidean distance. Instead, a polar coordinate representation could lead to a better result, as the data points in each string-shape cluster are close to each other, in terms of polar angle.

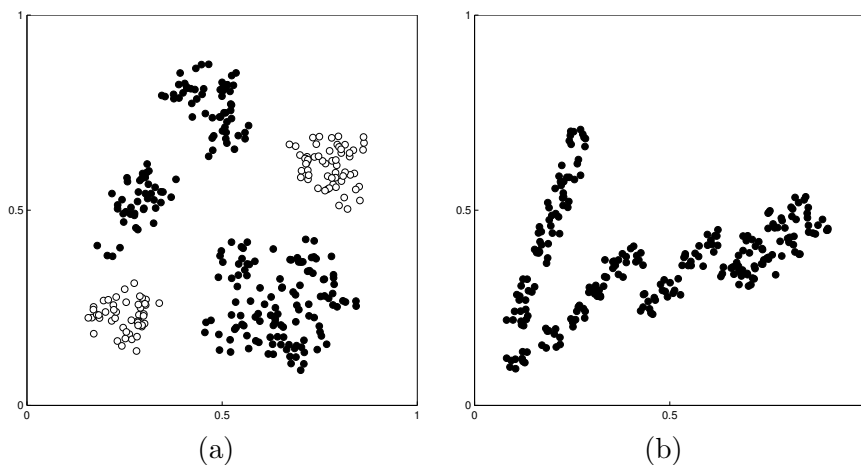


Figure 2: To identify compact clusters, a cartesian coordinate representation is more suitable for case (a), while a polar coordinate representation is more suitable for case (b).

Feature selection and extraction play an important role for abstracting complex patterns into a machine understandable representation. The feature set used by a clustering system regularizes the “area” that the system gives “attention” to. Referring to the data set in Figure 2a, if *coordinate position* is selected as the feature set, many clustering algorithms would be capable of identifying the five compact clusters. However, if only the *color* of the data

points is selected as the feature, a clustering system would probably output only two clusters, containing *white points* and *black points* respectively.

The feature set also affects the quality as well as the efficiency of a clustering system. A large feature set containing numerous irrelevant features does not improve the clustering quality but increases the computational complexity of the system. On the other hand, an insufficient feature set may decrease the accuracy of the representation and therefore cause potential loss of important patterns in the clustering output.

2.2 Pattern Proximity Measure

Pattern proximity refers to the metric that evaluates the similarity (or in contrast, the dissimilarity/distance) between two patterns. While a number of clustering methods (such as [17]) disclaim the use of specific distance measures, they use alternative pattern proximity measures to evaluate the so-called *relationship* between two patterns. A pattern proximity measure serves as the basis for cluster generation as it indicates how two patterns “look alike” to each other.

Since the type, the range, and the format of the input features are defined during the pattern representation stage, it follows that a pattern proximity measure should correspond to the pattern representation. In addition, a good proximity measure should be capable of utilizing only the key features of the data domain. Referring to Figure 2 again, with a cartesian representation, Euclidean distance is suitable to identify the geometric differences among the five clusters in data set (a) but may not be capable enough to recognize the clusters in data set (b). Instead, cosine distance is more suitable for data set (b), as it gives no weight to a vector’s radius and focuses on the differences of the vectors’ projections on the polar angle only. Generally, a careful review on the existing correlations among patterns helps to choose a suitable pattern similarity measure.

Given an existing pattern representation paradigm, a data set may be separable in various ways. Under this condition, using different pattern proximity measures may result in very different clustering outputs. Figure 3 depicts a simple example using eight speed cameras on three roads. Based on different proximity criteria listed in Table 1, there are different solutions for clustering the speed cameras, each with an acceptable interpretation. In most cases, the clustering system is desired to output only one (or a few number of) optimal grouping solution that best matches the user intention on the data set, although that may be partial and subjective. Hence it is

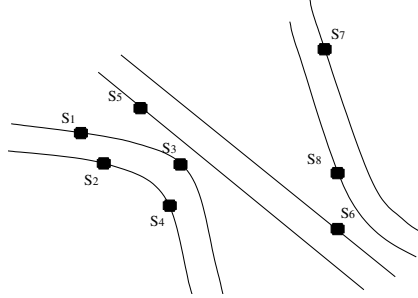


Figure 3: Using various pattern proximity measures, the eight speed cameras on the three roads may be clustered into different cluster groupings.

Table 1: Interpretations to the various clustering results of the eight speed cameras, based on different pattern proximity measures.

Measure	Clustering Result	Interpretation
Geometric distance	$C_1 = \{S_1, S_2, S_3, S_4, S_5\}$ $C_2 = \{S_6, S_8\}$ $C_3 = \{S_7\}$	Cameras in each cluster are geometrically closer to each other than to those in other clusters.
Connectivity	$C_1 = \{S_1, S_2, S_3, S_4\}$ $C_2 = \{S_5, S_6\}$ $C_3 = \{S_7, S_8\}$	Each cluster contains the cameras in the same road.
Density	$C_1 = \{S_1, S_2, S_3, S_4, S_5\}$ $C_2 = \{S_6, S_7, S_8\}$	C_1 identifies the zone intensively equipped with cameras, in contrast to the rest of the area.

important to identify the pattern proximity measure that effectively and precisely formulates the user’s intention on the patterns.

2.3 Clustering Algorithms

A clustering algorithm groups the input data according to a set of predefined criteria. The clustering algorithm used by a system can be either statistical or heuristic. In essence, the objective of clustering is to maximize the intra-cluster similarity and minimize the inter-cluster similarity [22]. A large variety of clustering algorithms have been extensively studied in the

literature. While a comprehensive survey of clustering algorithms is not the focus of our study, we give a bird’s-eye review of various types of the available algorithms in Table 2.

Table 2: Various types of clustering methods, based on learning paradigm, codebook size, cluster assignment, and system architecture respectively.

Criteria	Categories
Learning paradigm	<i>Off-line</i> : Iteratively batch learning on the whole input set. <i>On-line</i> : Incremental learning that does not remember the specific input history.
Codebook size (number of output clusters)	<i>Static-sizing</i> : The codebook size is fixed. <i>Dynamic-sizing</i> : The codebook size is adaptive to the distribution of input data.
Cluster assignment	<i>Hard</i> : Each input is assigned with one class label. <i>Fuzzy</i> : Each input is given a degree of membership with every output cluster.
System architecture	<i>Partitioning</i> : The input space is naively separated into disjoint output clusters. <i>Hierarchical</i> : The output tree shows the relations among clusters. <i>Density-based</i> : The input data are grouped based on density conditions. <i>Grid-based</i> : The spacial input space is quantized into finite sub-spaces (grids) before clustering of each sub-space.

Despite the numerous clustering algorithms available in the literature, there is no single method that can cope with all clustering problems. The choice of the clustering algorithm for a specific task affects the clustering result in a fundamental way. In addition, the learning activity of a large number of clustering algorithms is controlled and hence affected by a set of internal parameters. The optimal parameter set is usually decided through empirical experiments on the specific data set.

3 Evaluation of Clustering Quality

With the summarization of the various factors that affect the result of a clustering system, it is desirable for a clustering system to be capable of providing the necessary feedback to each stage of the clustering process, based on the evaluation and the interpretation of the clustering output. Such a feedback helps to gather prior knowledge on the data distribution, define a suitable pattern proximity measure that matches the problem domain, choose a clustering algorithm as well as decide the optimal parameter setting for the specific algorithm, and therefore lead to an optimal clustering output.

Human inspection on the clustering output may be the most intuitive clustering validation method as it compares the clustering result with the user’s intention in a natural way. In fact, human inspection has been widely used to validate the clustering outputs of the controlled experiments on two dimensional data sets. However, human inspection lacks the scalability to high dimensional, large, and complicated problem domains. In addition, manual inspection is always not desirable and not feasible in real-life applications. Therefore, quantitative assessment of clustering quality is of great importance for various clustering applications.

When talking about quantitative measure of the clustering quality, readers should be aware that the definition of the *quality* actually is quite subjective. Given an input data set, users may have varying desires on the knowledge that a clustering system could discover, and therefore give different interpretation of the clustering *quality*. Before a quality measure is applied to evaluate the clustering results generated by two different systems, a study on the *quantitative comparability* of the outputs is fundamentally important. We consider two clustering systems quantitative comparable only if:

- *the pattern representation of the two systems are similar enough,*
- *they are based on the similar clustering fundament, and*
- *they intend to fulfill the same user requirement, optionally to optimize the same criteria.*

We give explanations through examples below. Given the speed camera example as in Figure 3, the three clustering results in Table 1 are not quantitatively comparable as they are based on different clustering criteria. Likewise, given the data set as in Figure 2a, the clustering output based on

the *coordinate position* of each data point is not quantitatively comparable with that based on the *color* of each data point, as they are based on totally different feature sets. However, using the same feature representation (*coordinate position*), the outputs of the two clustering methods as in Figure 4 are quantitatively comparable as they both attempt to identify compact clusters of the data set.

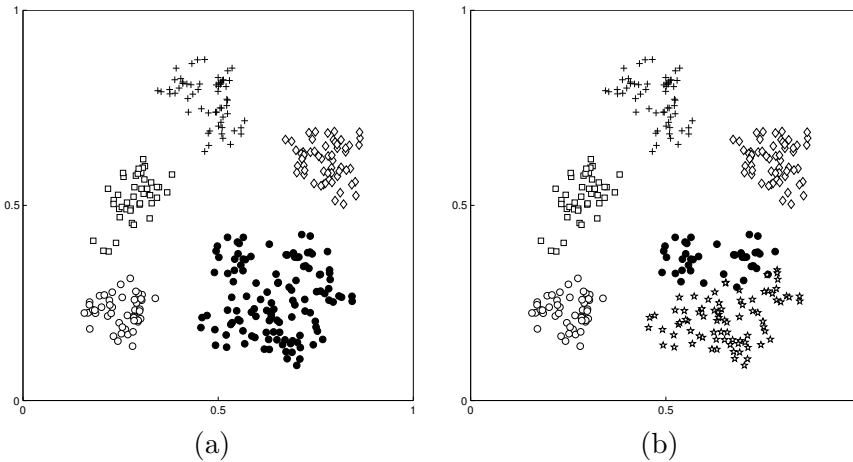


Figure 4: Two quantitatively comparable partitioning outputs of the data set in Figure 2a. Each type of marker identifies data points in the same cluster. Result (a) is considered to have a higher quality than result (b) in the sense that it recognizes the large cluster (marked with solid circles in (a)) more precisely.

A multitude of clustering evaluation measures have been extensively studied in the literature. Examples include the Dunn and Dunn-like family of measures initially proposed for evaluation of *crisp* [7], the variances of DB measures [6, 16], and the relatively recent SD validity indices [11]. However, due to the limitations mentioned above, a large number of these evaluation measures are capable of validating a narrow ranges of in-house clustering methods only. Our study intends to use a set of evaluation measures with high scalability, in terms of the capability for evaluating a wide range of clustering systems. In the rest of this section, we introduce several clustering evaluation measures based on two different statistical fundamentals, i.e. cluster distribution and class conformation respectively.

3.1 Evaluation Measures Based on Cluster Distribution

The objective of clustering has been widely quoted as to reorganize the input data set in an unsupervised way such that data points in the same cluster are more similar to each other than to points in different clusters. When explaining this objective in the quantitative manner, it is to minimize the distances among the data points in individual clusters and to maximize the distances between clusters. Therefore, it is a natural way to validate the intra-cluster homogeneity and the inter-cluster separation of the clustering output in a global fashion, using the quantities inherent to the distribution of the output data.

We extend our study from the various clustering validity methods in this category [9, 11], and propose two quality evaluation measures, namely: *cluster compactness* and *cluster separation*. The definitions of these measures are given as below.

Cluster Compactness: The *cluster compactness* measure introduced in our study, is based on our generalized definition of the *variance* of a vector data set given by

$$v(\mathbf{X}) = \sqrt{\frac{1}{N} \sum_{i=1}^N d^2(\mathbf{x}_i, \bar{\mathbf{x}})} \quad (1)$$

where $d(\mathbf{x}_i, \mathbf{x}_j)$ is a distance metric between two vectors \mathbf{x}_i and \mathbf{x}_j , N is the number of members in \mathbf{X} , and $\bar{\mathbf{x}} = \frac{1}{N} \sum_i \mathbf{x}_i$ is the mean of \mathbf{X} . A smaller *variance* value of a data set indicates a higher homogeneity of the vectors in the data set, in terms of the distance measure $d()$. Particularly, when \mathbf{X} is one-dimensional and $d()$ is the Euclidean distance, $v(\mathbf{X})$ becomes the statistical variance of the data set $\sigma(\mathbf{X})$. The *cluster compactness* for the output clusters $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_C$ generated by a system is then defined as

$$Cmp = \frac{1}{C} \sum_i^C \frac{v(\mathbf{c}_i)}{v(\mathbf{X})}, \quad (2)$$

where C is the number of clusters generated on the data set \mathbf{X} , $v(\mathbf{c}_i)$ is the variance of the cluster \mathbf{c}_i , and $v(\mathbf{X})$ is the variance of the data set \mathbf{X} .

The *cluster compactness* measure evaluates how well the subsets (output clusters) of the input is redistributed by the clustering system, compared with the whole input set, in terms of the data homogeneity reflected by the distance metric used by the clustering system. When the distance metric is the Euclidean distance, the *cluster compactness* measure becomes coherent

to the *average cluster scattering* index used in Halkidi et al’s study [11]. It is understandable that, for the *cluster compactness* measure, a smaller value indicates a higher average compactness in the output clusters. This however does not necessarily mean a “better” clustering output. Given a clustering system that encodes each and every unique input data into one separate cluster, the *cluster compactness* score of its output has a minimal value of 0. Such a clustering output is however not desirable. To tackle this, we introduce the *cluster separation* measure to complement the evaluation.

Cluster Separation: The *cluster separation* measure introduced here borrows the idea in [11] and combines the idea of the clustering evaluation function introduced by [9]. The *cluster separation* of a clustering system’s output is defined by

$$Sep = \frac{1}{C(C-1)} \sum_{i=1}^C \sum_{j=1, j \neq i}^C \exp\left(-\frac{d^2(\mathbf{x}_{c_i}, \mathbf{x}_{c_j})}{2\sigma^2}\right), \quad (3)$$

where σ is a Gaussian constant, C is the number of clusters, \mathbf{x}_{c_i} is the centroid of the cluster \mathbf{c}_i , $d()$ is the distance metric used by the clustering system, and $d(\mathbf{x}_{c_i}, \mathbf{x}_{c_j})$ is the distance between the centroid of \mathbf{c}_i and the centroid of \mathbf{c}_j .

It is noted that the pair-wise distances among the output cluster centroids are the key components of the *cluster separation* measure. The Gaussian function and the L1-normalization normalizes its value to between 0 and 1. A smaller *cluster separation* score indicates a larger overall dissimilarity among the output clusters. However, given the particular case that a clustering system output the whole input set into one cluster, the *cluster separation* score reaches to minimal value of 0, which is not applicably desirable. Hence, we reach the necessary point to combine the *cluster compactness* and *cluster separation* measures into one in order to tackle each one’s deficiency and evaluate the overall performance of a clustering system. An intuitive combination, named *overall cluster quality*, is defined as:

$$Ocq(\beta) = \beta \cdot Cmp + (1 - \beta) \cdot Sep, \quad (4)$$

where $\beta \in [0, 1]$ is the weight that balances measures *cluster compactness* and *cluster separation*. Particularly, $Ocq(0.5)$ gives equal weights to the two measures. Readers however should be aware of the limitation of this combination: although the combination measure Ocq facilitates the comparison work, one may find it not easy to interpret the combined value, as the Cmp and Sep scores are not measured in the same dimension. In addition, in some

other cases, evaluating a clustering system through intra-cluster compactness and inter-cluster separation respectively helps to gain more insightful understanding of the system characteristics.

3.2 Evaluation Measures Based on Class Conformation

This category of validation measures assumes that there is a desirable distribution of the data set with which it is possible to perform a direct comparison of the clustering output. Following the data distribution, one can assign a class label to each data point. The target of the clustering system can then be correspondingly interpreted as to replicate the underlying class structure through unsupervised learning. In an optimal clustering output, data points with the same class labels are clustered into the same cluster and data points with different class labels appear in different clusters. We describe two quantitative evaluation measures based on these criteria as follows.

Cluster Entropy: Boley [1] introduced an information entropy approach to evaluate the quality of a set of clusters according to the original class labels of the data points. For each cluster \mathbf{c}_i , a cluster entropy Ec_i is computed by

$$Ec_i = - \sum_j \frac{n(l_j, c_i)}{n(c_i)} \log \frac{n(l_j, c_i)}{n(c_i)} \quad (5)$$

where $n(l_j, c_i)$ is the number of the samples in cluster \mathbf{c}_i with a predefined label l_j and $n(c_i) = \sum_j n(l_j, c_i)$ is the number of samples in cluster \mathbf{c}_i . The overall *cluster entropy* Ec is then given by a weighted sum of individual cluster entropies by

$$Ec = \frac{1}{\sum_i n(c_i)} \sum_i n(c_i) Ec_i. \quad (6)$$

The cluster entropy reflects the quality of individual clusters in terms of homogeneity of the data points in a cluster (a smaller value indicates a higher homogeneity). It however does not measure the compactness of a clustering solution in terms of the number of clusters generated. A clustering system that generates many clusters would tend to have very low cluster entropies but is not necessarily desirable. To counter this deficiency, we use another entropy measure below to measure how data points of the same class are represented by the various clusters created.

Class Entropy: For each class \mathbf{l}_j , a class entropy El_j is computed by

$$El_j = - \sum_i \frac{n(l_j, c_i)}{n(l_j)} \log \frac{n(l_j, c_i)}{n(l_j)} \quad (7)$$

where $n(l_j, c_i)$ is the number of samples in cluster \mathbf{c}_i with a predefined label l_j and $n(l_j) = \sum_i n(l_j, c_i)$ is the number of the samples with class label l_j . The overall *class entropy* El is then given by a weighted sum of individual class entropies by

$$El = \frac{1}{\sum_j n(l_j)} \sum_j n(l_j) El_j. \quad (8)$$

Since both the *cluster entropy* and *class entropy* utilize the predefined class labels on the input data only, they are independent to the choice of the feature representation and the pattern proximity measure. Therefore these two measures are practically capable of evaluating any clustering system. Compared with the cluster distribution based measures, these class conformation based measures are likely to have more advantages for identifying the optical clustering solution to match the user’s intention on the problem domain. One apparent drawback however is the potential complexity of the labelling process, which may not be feasible and not desirable in real-life applications.

Our prior study showed that, similar to the characteristics of the *cluster compactness* and the *cluster separation* measures, as the number of clusters over one data set increases, the *cluster entropy* generally decreases while the *class entropy* increases. We follow the identical paradigm for the combination of *cluster compactness* and *cluster separation*, and define a combined *overall entropy* measure:

$$Ecl(\beta) = \beta \cdot Ec + (1 - \beta) \cdot El, \quad (9)$$

where $\beta \in [0, 1]$ is the weight that balances the two measures.

4 Clustering Methods

Before introducing our experiments on the various clustering quality measures, review of the clustering systems used in our experiments helps toward a better understanding of the experimental results. Our experiments tested three partitioning methods that work with a fixed number of clusters, namely k-means [18], Self-Organizing Maps (SOM) [15], and Adaptive Resonance

Theory under Constraints (ART-C) [12]. The learning algorithms of the three methods are summarized as follows.

4.1 k-means

k-means [18] has been extensively studied and applied in the clustering literature due to its simplicity and robustness. The fundament of the k-means clustering method is to minimize the intra-cluster compactness of the output, in terms of the summed squared error. The k-means clustering paradigm is summarized below.

1. Initialize the k reference clusters with randomly chosen input points or through certain estimation of the data distribution.
2. Assign each input point to the nearest cluster centroid.
3. Recalculate the centroid of each cluster using the mean of the input points in the cluster.
4. Repeat from step 2 until convergence.

Since k-means follows a batch learning paradigm to iteratively adjust the cluster centroids, it is inefficient in handling large scale data sets. In addition, the output of k-means is affected by the cluster initialization method. Its strength however lies in the satisfactory quality in the sense that the output has locally minimal summed squared error when it converges.

4.2 Self-Organizing Maps (SOM)

SOM as proposed by Kohonen [15] is a family of self-organizing neural networks widely used for clustering and visualization. As an unique feature of SOM, the clusters in a SOM network are organized in a multi-dimensional map. During learning, the network updates not only the winner's weight, but also the weights of the winner's neighbors. This results in an output map with a distribution that similar patterns (clusters) are placed together. The learning paradigm of SOM is given below.

1. Initialize each reference cluster $\mathbf{w}_j^{(0)}$ with random values. Set the initial neighborhood set of each cluster $N_j^{(0)}$ to be large.
2. Given an input \mathbf{x}_i , find the winner node J that has the maximal similarity with \mathbf{x}_i .

3. Update the cluster vectors of the winner node and its neighbors, according to

$$\mathbf{w}_j^{(t+1)} = \mathbf{w}_j^{(t)} + \eta^{(t)} h(j, J) (\mathbf{x}_i - \mathbf{w}_j^{(t)}) \text{ for each } j \in N_j^{(t)}, \quad (10)$$

where $h(j, J) \in [0, 1]$ is a scalar *kernel* function that gives a higher weight to a closer neighbor of the winner node J and $\eta^{(t)}$ is the learning rate.

4. At the end of the learning iteration, shrink the neighborhood sets so that $N_j^{(t+1)} \subset N_j^{(t)}$ for each j and decrease the learning rate so that $\eta^{(t+1)} < \eta^{(t)}$. Repeat from step 2 until convergence.

There are a large number of SOM variances depending on the dimension of the organization map, the definition of the neighborhood sets N_i , the kernel function $h()$, as well as the paradigm that iteratively readjusts N_i and η . When the network utilizes one-dimensional map and the degree of neighborhood is zero (i.e. the neighborhood contains the winner node only), SOM equals to an online learning variance of the k-means clustering method [8]. It is also understandable that, if the neighborhood degree shrinks to zero at the end of the learning, SOM is capable of obtaining optimal output with locally minimal summed square error. The drawback of SOM is that its learning activity is affected by the initialization of network and the presentation order of the input data.

4.3 Adaptive Resonance Theory under Constraints (ART-C)

ART-C [12] is a new variance of the Adaptive Resonance Theory (ART) neural networks, which was originally developed by Carpenter and Grossberg [5]. Unlike the conventional ART modules which work on a dynamic number of output clusters, an ART-C module is capable of satisfying a user *constraint* on its codebook size (i.e. the number of output clusters), while keeping the stability-plastically of ART intact. Compared with the conventional ART architecture, the ART-C architecture (Figure 5) contains an added constraining subsystem, which interacts with the ART's attentional subsystem and the orienting subsystem. During learning, the constraining subsystem adaptively estimates the distribution of the input data and self-adjusts the vigilance parameter for the orienting subsystem, which in turn governs the learning activities in the attentional subsystem. The learning paradigm of ART-C is summarized below.

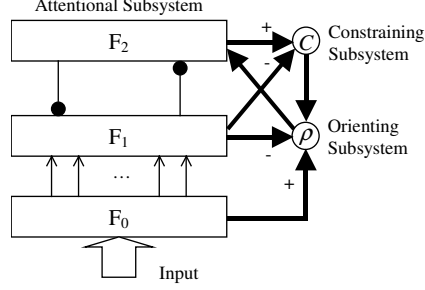


Figure 5: The ART-C architecture.

1. Initialize the network's recognition layer F_2 with null set \emptyset (i.e. the number of clusters in F_2 is zero) and set the network's *vigilance* ρ as 1.0.
2. Given an input \mathbf{x}_i in the input layer F_0 , the comparison layer F_1 stores the *match* scores $M(\mathbf{x}_i, \mathbf{w}_j)$ between the input and every cluster vector \mathbf{w}_j in F_2 .
3. If the maximal *match* scores satisfies $\max\{M(\mathbf{x}_i, \mathbf{w}_j)\} \geq \rho$ or the number of clusters c in F_2 satisfies $c < C$, where C is the user constraint on the number of output clusters, then carry out the *conventional ART learning* process, which is summarized below,
 - (a) Calculate the *choice* scores $T(\mathbf{x}_i, \mathbf{w}_j)$ between the input and every cluster vector \mathbf{w}_j in F_2 .
 - (b) Identify the winner node J that receives the maximal *choice* score $T(\mathbf{x}_i, \mathbf{w}_J)$. *Resonance* happens when the *match* score $M(\mathbf{x}_i, \mathbf{w}_J) \geq \rho$. Otherwise *reset* the winner node J and repeat the search process.
 - (c) If the network reaches *resonance*, the network updates the cluster vector of the winner node, according to a learning function

$$\mathbf{w}_J^{(t+1)} = L(\mathbf{x}_i, \mathbf{w}_J^{(t)}). \quad (11)$$

Otherwise, if all F_2 nodes j are reset, insert the input \mathbf{x}_i into the F_2 layer as a new reference cluster.

otherwise, do *constraint reset*, which is summarized below,

- (a) Insert input \mathbf{x}_i into F_2 layer as a new reference cluster.
- (b) Calculate the pair wise *match* score $M(\mathbf{w}_i, \mathbf{w}_j)$ of every F_2 node pairs.
- (c) Locate the winner pair (I, J) according to

$$M(\mathbf{w}_I, \mathbf{w}_J) = \max\{M(\mathbf{w}_i, \mathbf{w}_j) : i \in \mathbf{R}\}, \quad (12)$$

where \mathbf{R} is the set of F_2 nodes indexed by the criteria:

$$\mathbf{R} = \{F_2 \text{ node } i \text{ whose } \max\{M(\mathbf{w}_i, \mathbf{w}_j) : j \neq i\} < \rho\}. \quad (13)$$

- (d) Modify the network's *vigilance* according to the *match* score between the winner pair:

$$\rho^{new} = M(\mathbf{w}_I, \mathbf{w}_J). \quad (14)$$

- (e) Update \mathbf{w}_J with \mathbf{w}_I , by utilizing the ART *learning* function:

$$\mathbf{w}_J^{(t+1)} = L(\mathbf{w}_I^{(t)}, \mathbf{w}_J^{(t)}). \quad (15)$$

- (f) Delete node I from F_2 layer.

4. Repeat from step 2 until convergence.

The ART module used in the ART-C architecture can be either ART-1 [5], ART-2 [2, 4] or fuzzy ART [3], each using a different set of *choice*, *match*, and *learning* functions. ART-2 utilizes the cosine similarity as the *choice* and *match* functions:

$$T(\mathbf{x}_i, \mathbf{w}_j) = M(\mathbf{x}_i, \mathbf{w}_j) = \frac{\mathbf{x}_i \cdot \mathbf{w}_j}{\|\mathbf{x}_i\| \|\mathbf{w}_j\|}, \quad (16)$$

where the L_2 -norm function $\|\cdot\|$ is defined by

$$\|\mathbf{x}\| = \sqrt{\sum_i x_i^2} \quad (17)$$

for vector \mathbf{x} . The learning function is given by

$$L(\mathbf{x}_i, \mathbf{w}_j) = \eta \mathbf{x}_i + (1 - \eta) \mathbf{w}_j. \quad (18)$$

where η is the learning rate. As a comparison, fuzzy ART uses hyper-rectangle based *choice* and *match* functions

$$T(\mathbf{x}_i, \mathbf{w}_j) = \frac{|\mathbf{x}_i \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|}, \quad (19)$$

$$M(\mathbf{x}_i, \mathbf{w}_j) = \frac{|\mathbf{x}_i \wedge \mathbf{w}_j|}{|\mathbf{x}_i|}, \quad (20)$$

where α is a constant, the fuzzy AND operation \wedge is defined by

$$(\mathbf{p} \wedge \mathbf{q})_i \equiv \min(p_i, q_i) \quad (21)$$

and the L1-norm $|\cdot|$ is defined by

$$|\mathbf{p}| \equiv \sum_i p_i \quad (22)$$

for vectors \mathbf{p} and \mathbf{q} . The learning function for fuzzy ART is given by

$$L(\mathbf{x}_i, \mathbf{w}_j) = \eta(\mathbf{x}_i \wedge \mathbf{w}_j) + (1 - \eta)\mathbf{w}_j. \quad (23)$$

ART-C adaptively generates reference clusters (recognition categories) using the input samples. Therefore no prior knowledge on the data distribution is required for the initialization of the network. However, like most online learning algorithms, ART-C's learning is affected by the presentation order of the input data.

5 Experiments and Discussions

The experiments reported in this section illustrate the use of the various clustering evaluation measures for a variety of purposes. We extend our discussion on the two synthetic data sets as in Figure 2a and 2b in a quantitative manner. Tasks on these two data sets include identifying the optimal number of clusters in the data set and choosing of a pattern proximity measure suitable for the specific data distribution. In addition, using a high-dimensional and sparse real-life data set, i.e. the Reuters-21578 free text collection, we carry out comparisons across the performance of three distinct clustering algorithms, namely k-means, SOM, and ART-C, to discover the similarities and differences of their clustering behaviors.

5.1 Statistical Validation of Comparative Observation

It is noted that, for both k-means and SOM, the clustering output is affected by the initialization of cluster prototypes. In addition, the order of the input sequence affect the outputs of SOM and ART-C. Therefore, comparative findings based on a single experiment would not be representative due to the potential deviation of the observation values.

To tackle this deficiency, we adopted the commonly used statistical validation paradigm in our experiments. Given a clustering task, we repeat the experiments for each clustering method under evaluation for ten times. In each experiment, the presenting sequence of the input data is reshuffled and the clustering methods are trained to convergence. Based on the observation values from the ten runs, the *means* and the *standard deviations* are reported. In order to compare the evaluation scores obtained, we compare the *mean* values and employed t-test to validate the significance of the comparative observations across the ten runs.

5.2 Identification of the Optimal Number of Clusters

Our experiments on the synthetic data set as shown in Figure 2a evaluated the k-means clustering method using the Euclidean distance. The synthetic data set contains 334 data points, each is a 2-dimensional vector of values between 0 and 1. Our task is to identify the optimal number of clusters on the data set in an unsupervised way. Here the *optimal* solution refers to the result that best reflects the data distribution and matches the user’s post-validation on the data set, in terms of intra-cluster compactness and inter-cluster separation of the output clusters.

The paradigm of the experiment is summarized below. We apply the k-means method on the data set using k values ranging from 2 to 8. For each k value, we evaluate the quality of the output using the measures based on *cluster compactness* (Cmp) and *cluster separation* (Sep). This enables us to observe the change of the score values according to the change of k . Intuitively the most satisfactory quality score indicates the best partition of the data set, while the corresponding k value suggests the optimal number of clusters on the data set.

Figure 6 depicts the change of *cluster compactness*, *cluster separation*, as well as *overall cluster quality* by varying k from 2 to 8. $2\sigma^2 = 0.25$ is used for the ease of evaluation in Equation 3 and $\beta = 0.5$ for $Ocq()$ is used to give equal weights to *cluster compactness* and *cluster separation*. To

obtain a clear illustration, only the mean values of the ten observations over each measure are plotted in the figure while the standard deviations are not reported.

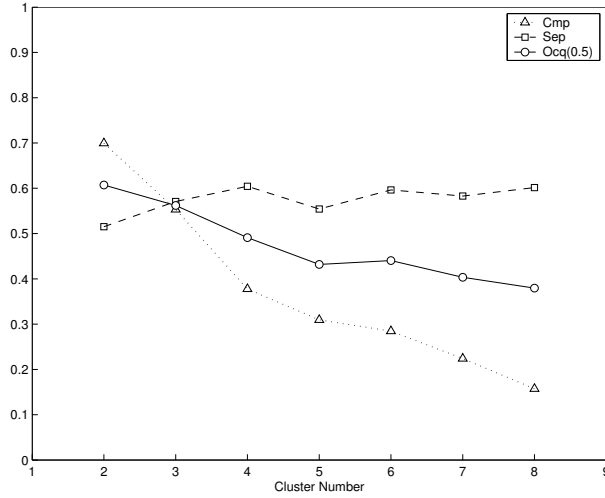


Figure 6: *Cluster compactness, cluster separation, and overall cluster quality* of k-means on the synthetic data set in Figure 2a. The locally minimal value of *overall cluster quality* $Ocq(0.5)$ at $k = 5$ suggests the optimal number of clusters on the data set.

It is noted that, when k increases, *cluster compactness* gradiently decreases and *cluster separation* generally increases. This is due to the nature that a larger number of partitions on the same data space generally tends to decrease the size of each partition (which causes higher compactness in each partition) as well as the distances among the partition centroids (which causes lower separation of partitions). However, as an apparent exception, the *cluster separation* shows a locally minimal value at $k = 5$, while the decreasing trend of *cluster separation* at $k = 5$ is significantly different from those at different k values. The *overall cluster quality* also shows a locally minimal value at $k = 5$. This suggests that the optimal number of clusters (in terms of Euclidean similarity) is five. The result is supported by human inspection of the data in Figure 2a.

The drawbacks of this experimental paradigm however are notable. First, it is not easy to suggest a proper range of k values for the iterative testing if the user lacks a prior estimation of the data distribution. In addition, both

the σ value for the calculation of *cluster separation* and the weight β for the calculation of *overall cluster quality* are subjectively determined. This shows that human interaction and prior knowledge on the problem domain is still required for the use of these evaluation measures.

5.3 Selection of Pattern Proximity Measure

Our experiments on the synthetic data set as shown in Figure 2b utilize the quality measures based on class conformation to evaluate two variances of ART-C networks, namely fuzzy ART-C (based on Fuzzy ART) and ART2-C (based on ART-2). While fuzzy ART-C groups input data according to *nearest hyper-rectangles*, ART2-C groups input data according to *nearest neighbors*, in terms of cosine similarity. Our comparative experiments attempt to discover which of these two pattern proximity measures is more capable of identifying the data distribution on the problem domain, and therefore produces clustering output with a higher match with the user intent.

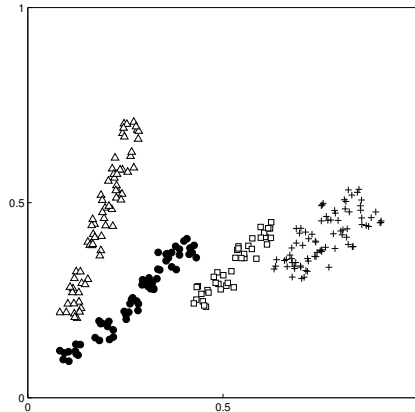


Figure 7: The manually labelled data set as in Figure 2b. Data points assigned with the same class label are identified with the same marker.

In order to evaluate the clustering quality using *cluster entropy*, *class entropy*, and *overall entropy*, we pre-assigned each data point with a class label based on our observation. There are four different class labels assigned to the 250 data points in the collection, each label corresponding to a string-shaped class in Figure 7.

Our experiments compared fuzzy ART-C and ART2-C with a preset

constraint (C) of 4, each using a standard set of parameter values. Table 3 summarizes the statistics of the comparison results. While ART2-C is capable of producing a better balanced set of *cluster entropy* and *class entropy* scores (which indicates a better balance of cluster homogeneity and class compactness), the *cluster entropy* score of fuzzy ART-C is three times higher than that of ART2-C in our experiment. Although the *class entropy* score of fuzzy ART-C is slightly lower than that of ART2-C, the weighted *overall entropy* $Ecl(0.5)$ of fuzzy ART-C is significantly higher than that of ART2-C due to the high *cluster entropy* value. This indicates that the cosine similarity based paradigm is more suitable than the nearest hyper-rectangle based paradigm on the tested problem domain. This result is not surprising to us, as prior comparison studies on hyper-rectangle methods also showed that they perform well only when the data boundaries are roughly parallel to the coordinates axes [19].

Table 3: *Cluster entropy*, *class entropy*, and *overall entropy* of ART2-C and fuzzy ART-C on the synthetic data set in Figure 7. Both methods work on $C = 4$. All values are shown with the means and the standard deviations over ten runs.

Method	Ec	El	$Ecl(0.5)$
ART2-C	0.1483 ± 0.0173	0.1142 ± 0.0225	0.1312 ± 0.0092
fuzzy ART-C	0.5800 ± 0.0037	0.0337 ± 0.0064	0.3068 ± 0.0020

5.4 Cross-Comparison of Clustering Methods

We applied the four evaluation measures introduced in this chapter, namely *cluster compactness* (Cmp), *cluster separation* (Sep), *cluster entropy* (Ec), and *class entropy* (El) to compare the performance of k-means, Self-Organizing Maps (SOM), and Adaptive Resonance Theory under Constraints (ART-C) on a sparse and high-dimensional real-life data set, namely the Reuters-21578 free text collection. The details of our benchmark study are reported in the following subsections.

5.4.1 Data Preparation

The Reuters-21578 data set is a collection of documents that appeared on the Reuters news-wire in 1987. Since the data set was originally released for the

evaluation of text classification systems, the documents have been carefully assembled and indexed with class labels by personnel from Reuters Ltd. Our experiments used the subset of documents from the top ten categories. To facilitate our evaluation, documents that were originally indexed with multiple class labels were duplicated in our experiment so that each copy was associated with one class label.

We adopted the bag-of-words feature representation scheme for the documents. *CHI* (χ) statistics [21] was employed as the ranking metric for feature selection. Based on a bag of 335 top-ranking keyword features, the content of each document was represented as an in-document term frequency (TF) vector, which was then processed using an inverse document frequency (IDF) based weighting method [20] and subsequently L2-normalized. After the removal of 57 null vectors (i.e. vectors with all attributes equal to 0), we obtained a set of 9,968 labelled vectors for our experimental study.

5.4.2 Evaluation Paradigm

All three methods used the cosine similarity measure and a standard set of parameter values. As the data collection is large and sparse, we did not expect the clustering methods to replicate the ten clusters corresponding to the ten original classes. Instead, we carried out two set of experiments, each setting the number of the output clusters to be 25 and 81 respectively. In the SOM architecture, these correspond to a 5 by 5, and a 9 by 9 two-dimensional maps.

Unlike the previous two experiments, our experiment on the Reuters-21578 data set evaluated the three clustering methods' performance using each measure separately for a better understanding of their properties. $2\sigma^2 = 1.0$ was used for the ease of computing *cluster separation*. Since the document vectors are preprocessed with L2-normalization, we use Euclidean distance for the evaluation of *cluster compactness* and *cluster separation*. This is due to the high correlation between the cosine similarity and the Euclidean distance under this condition, i.e. high cosine similarity corresponds to close Euclidean distance.

In addition to the four cluster quality measures, the time complexity of each tested system, in terms of the CPU time used on each experiment, were reported and compared. Based on these, we empirically examined the learning efficiency of each method. To facilitate the comparison, all the three systems were implemented with C++ programs that shared a common set of functions for vector manipulation.

5.4.3 Results and Discussions

Table 4 reports the experimental results on k-means, SOM, and ART2-C. Working with 81 clusters, the output of SOM showed a slightly worse quality than that of k-means, in terms of *cluster compactness*, *cluster separation*, and *class entropy*. It may be that the nature of the SOM’s learning in maintaining the neighborhood relationship decreases the dissimilarity among clusters as well as the compactness of each cluster. However, the differences were not significantly reflected when the number of output clusters was 25. In general, the evaluation scores of k-means and SOM were rather similar, compared with those of ART2-C. This may be due to the same paradigm we use to initialize the cluster prototypes for both k-means and SOM. More importantly, the learning paradigms of k-means and SOM are more similar to each other than to that of ART2-C. We thus focus our further discussions on the comparison of ART2-C with k-means and SOM.

Table 4: Experimental results for k-means, SOM, and ART2-C on the Reuters-21578 corpus, when the number of clusters were set to 25 and 81 respectively. *I* and *T* stand for the number of learning iterations and the cost of training time (in seconds) respectively. *Cmp*, *Sep*, *Ec* and *El* stand for cluster compactness, cluster separation, cluster entropy and class entropy respectively. All values are shown with the mean and the standard deviation over ten runs.

Cluster number = 25			
	k-means	SOM	ART2-C
<i>I</i>	10.9 ± 2.0	11.2 ± 1.8	2.2 ± 0.4
<i>T</i> (s)	103.36 ± 36.543	90.018 ± 13.738	42.724 ± 15.537
<i>Cmp</i>	0.4681 ± 0.0086	0.4560 ± 0.0106	0.5187 ± 0.0044
<i>Sep</i>	0.2312 ± 0.0286	0.2248 ± 0.0617	0.2063 ± 0.0143
<i>Ec</i>	0.2028 ± 0.0057	0.2154 ± 0.0122	0.2670 ± 0.0165
<i>El</i>	0.7795 ± 0.0056	0.7838 ± 0.0164	0.7586 ± 0.0341
Cluster number = 81			
	k-means	SOM	ART2-C
<i>I</i>	11.8 ± 1.6	12.3 ± 1.3	2.8 ± 1.0
<i>T</i> (s)	323.85 ± 74.285	310.84 ± 33.656	88.057 ± 45.738
<i>Cmp</i>	0.3957 ± 0.0116	0.4126 ± 0.0041	0.4594 ± 0.0037
<i>Sep</i>	0.1968 ± 0.0187	0.2135 ± 0.0217	0.1874 ± 0.0243
<i>Ec</i>	0.1808 ± 0.0019	0.1806 ± 0.0030	0.1983 ± 0.0060
<i>El</i>	1.2375 ± 0.0122	1.2592 ± 0.0065	1.1580 ± 0.0127

In our experiments, the *cluster entropy* scores (Ec) and the *cluster compactness* scores (Cmp) of ART2-C outputs were generally higher (which indicated worse data homogeneity within clusters) than those of SOM and k-means. Our explanations are as follows: The learning paradigms of SOM and k-means minimize the mean square error of the data points within the individual clusters. Therefore both SOM and k-means are more capable of generating compact clusters. In terms of *class entropy* (El), and *cluster separation* (Sep), the outputs of ART2-C were better than those of SOM and k-means. It may be that, whereas SOM and k-means modify existing, randomly initialized cluster prototypes to encode new samples, ART adaptively inserts recognition categories to encode new input samples that are significantly distinct from existing prototypes. Therefore, ART2-C tends to generate reference prototypes using distinct samples, which in turn makes the output clusters to be more dissimilar to each other. This unique neuron initialization mechanism appeared to be effective in representing diverse data patterns in the input set. Our observations were supported by the t-test validations in Table 5.

Table 5: Statistical significance of our cross-method comparisons between k-means, SOM, and ART2-C on the Reuters-21578 corpus. “>>” and “>” (or “<<” and “<”) denote the left-side value is greater (or smaller) than the right-side value at significance level 0.01 and 0.05 respectively.

Cluster number = 25					
	T	Cmp	Sep	Ec	El
ART2-C vs. SOM	<	>>	<<	>>	<
ART2-C vs. k-means	<<	>>	<	>>	<
Cluster number = 81					
	T	Cmp	Sep	Ec	El
ART2-C vs. SOM	<<	>>	<	>>	<<
ART2-C vs. k-means	<<	>>	<<	>>	<<

It is interesting that, since the two categories of quality measures are based on the same theoretical fundament (i.e. to encourage intra-cluster compactness and inter-cluster separation), the comparison results using the two categories of measures generally support each other. For an experiment that produces a low *cluster compactness* score, the *cluster entropy* score is accordingly low. For an experiment that produces a low *cluster separation* score, the *class entropy* score is generally low as well.

Besides the clustering quality of the trio, we are particularly interested in the learning efficiency of each method. k-means utilizes a relatively slow off-line learning paradigm in modifying cluster centroids to achieve minimal mean errors within clusters iteratively. SOM updates the winner's neighbors during the encoding of each input, which could be rather computationally expensive. The time cost of SOM's learning was generally comparable to that of k-means in our experiments. The learning paradigm of ART guarantees that only input data with significant similarity are to be encoded together. Therefore ART2-C is capable of encoding the inputs using a relatively high learning rate without causing oscillation. With this architectural advantage, ART2-C showed a promising learning efficiency in our experiments. The number of learning iterations used by ART-C was generally four to six times less than those used by SOM and k-means. The training time of ART-C was significantly less than those of SOM and k-means. This suggested the strength of ART-C in handling massive real-life data on the fly.

Based on the discussions above, we summarize our comparative findings below:

- While the clustering output of SOM showed a slightly worse quality than that of k-means in some of our experiments, the performances of k-means and SOM are roughly comparable.
- The intra-cluster compactness of ART2-C's output is generally worse than those of k-means and SOM's outputs.
- The inter-cluster separation of ART2-C's output is generally better than those of k-means and SOM's outputs.
- ART2-C shows a significantly higher efficiency than k-means and SOM.

6 Conclusions

Clustering is one of the major learning tasks in data mining research. The fundament of the clustering task is to group the input data into clusters such that data points in the same cluster are more similar to each other than to points in other clusters.

A clustering system consists of three major components, namely the feature representation, the definition of pattern proximity measure, and the grouping paradigm. This chapter gives insightful illustration of the various

factors in each aspect that affect the output of a clustering system. Our review work concludes that cluster analysis and interpretation serves as an important feedback for selecting the parameters in each stage towards an optimal clustering output.

While the evaluation of the clustering result may be subjective, quantitative assessment of the clustering quality is of great value for various research areas. Since the objective of a clustering method is to maximize the intra-cluster compactness as well as the inter-cluster separation, a multitude of evaluation measures based on these two principles have been extensively studied in the literature. This chapter reviews and introduces two sets of evaluation measures, namely evaluation measures based on cluster distribution and evaluation measures based on class conformation. Our experiments showed the strength of these evaluation measures in various tasks, including discovering the inherent data distribution for suggesting the optimal number of clusters, choosing a suitable pattern proximity measure for a problem domain, and comparing various clustering methods for a better understanding of their learning characteristics.

References

- [1] D. Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):325–344, 1998.
- [2] G.A. Carpenter, S. Grossberg, and D.B. Rosen. ART 2-A: An adaptive resonance algorithm for rapid category learning and recognition. *Neural Networks*, 4:493–504, 1991.
- [3] G.A. Carpenter, S. Grossberg, and D.B. Rosen. Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4:759–771, 1991.
- [4] G.A. Carpenter and S. Grossberg. ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, 26:4919–4930, 1987.
- [5] G.A. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image processing*, 34:54–115, 1987.
- [6] D.L. Davis and D.W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2), 1979.

- [7] J.C. Dunn. Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics*, 4:95–104, 1974.
- [8] A. Flexer. On the use of self-organizing maps for clustering and visualization. In *Proceedings of the 3rd European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, pages 80–88, 1999.
- [9] E. Gokcay and J.C. Principe. A new clustering evaluation function using Renyi’s information potential. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2000.
- [10] M. Halkidi and M. Vazirgiannis. A data set oriented approach for clustering algorithm selection. In *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, 2001.
- [11] M. Halkidi, M. Vazirgiannis, and I. Batistakis. Quality scheme assessment in the clustering process. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, pages 265–276, 2000.
- [12] J. He, A.H. Tan, and C.L. Tan. ART-C: A neural architecture for self-organization under constraints. In *Proceedings of International Joint Conference on Neural Networks (IJCNN)*, pages 2550–2555, 2002.
- [13] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Upper Saddle River, NJ, 1988.
- [14] A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.
- [15] T. Kohonen. *Self-Organization and Associative Memory*. Springer-Verlag, Berlin, second edition, 1997.
- [16] N.R. Pal and J. Biswas. Cluster validation using graph theoretic concepts. *Pattern Recognition*, 30(6), 1997.
- [17] G.D. Ramkumar and Arun N. Swami. Clustering data without distance functions. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 21(1):9–14, 1998.
- [18] J.T. Tou and R.C. Gonzalez. *Pattern Recognition Principles*. Addison-Wesley, Massachusetts, 1974.

- [19] D. Wettschereck and T.G. Dietterich. An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms. *Machine Learning*, 19(1):5–27, 1995.
- [20] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 42–49, 1999.
- [21] Y. Yang and J.P. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML)*, pages 412–420, 1997.
- [22] O.R. Zaiane, A. Foss, C.H. Lee, and W. Wang. On data clustering analysis: Scalability, constraints, and validation. In *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 28–39, 2002.