# Handbook of Cluster Analysis (provisional top level file)

C. Hennig, M. Meila, F. Murtagh, R. Rocci (eds.)

September 7, 2013

# Contents

# Chapter 1

# A Survey of Consensus Clustering

Joydeep Ghosh and Ayan Acharya

Department of ECE, University of Texas at Austin

**Abstract**

This chapter describes the problem of combining multiple partitionings of a set of objects into a single consolidated clustering without accessing the features or algorithms that determine these partitionings – popularly known as the problem of "consensus clustering". We illustrate different algorithms for solving the consensus clustering problem. The notion of dissimilarity between a pair of clustering solutions plays a key role in designing any cluster ensemble algorithm and a summary of such dissimilarity measures is also provided. We also cover recent efforts on combining classifier and clustering ensembles, leading to new approaches for semi-supervised learning and transfer learning. Finally, we describe several applications of consensus clustering.

## 1.1 Introduction

The design of multiple classifier systems to solve difficult classification problems, using techniques such as bagging, boosting and output combining [57, 65, 40, 38], has resulted in some of the most notable advances in classifier design over the past two decades. A popular ap-

proach is to train multiple "base" classifiers, whose outputs are combined to form a classifier ensemble. A survey of such ensemble techniques — including applications of them to many difficult real-world problems such as remote sensing, person recognition, one vs. all recognition, and medicine — can be found in [54]. Concurrently, analytical frameworks have been developed that quantify the improvements in classification results due to combining multiple models [64]. The extensive literature on the subject has shown that the ensemble created from independent, diversified classifiers, is usually more accurate as well as more reliable than its individual components, i.e., the base classifiers.

The demonstrated success of classifier ensembles provides a direct motivation to study effective ways of combining multiple clustering solutions as well. This chapter covers the theory, design and application of *cluster ensembles*, which combine multiple *"base clusterings"* of the same set of objects into a single consolidated clustering. Each base clustering refers to a *grouping* of the same set of objects or its transformed (or perturbed) version using a suitable clustering algorithm. The consolidated clustering is often referred to as the *consensus* solution. At first glance, this problem sounds similar to the problem of designing classifier ensembles. However, combining multiple clusterings poses additional challenges. First, the number of clusters produced may differ across the different *base* solutions [7]. The appropriate number of clusters in the consensus is also not known in advance and may depend on the scale at which the data is inspected. Moreover, cluster labels are symbolic and thus aligning cluster labels across different solutions requires solving a potentially difficult correspondence problem. Also, in the typical formulation,[1] the original data used to yield the base solutions are not available to the consensus mechanism, which has only access to the sets of cluster labels. In some schemes, one does have control on how the base clusterings are produced [23], while in others even this is not granted in order to allow applications involving knowledge reuse [59], as described later. Despite these added complications, cluster ensembles are inviting since typically, the variations in quality across a variety of clustering algorithms applied to a specific dataset tends to be more than the typical variation in accuracies returned by a collection of reasonable classifiers. This suggests that cluster ensembles may achieve greater improvements over the base solutions, when compared with ensembles

[1] In this paper, we shall not consider approaches where the feature values of the original data or of the cluster representatives are available to the consensus mechanism, e.g. [32]

of classifiers [26].

In fact, the potential motivations for using cluster ensembles are much broader than those for using classification or regression ensembles, where one is primarily interested in improving predictive accuracy. These reasons include:

(a) **Improved Quality of Solution**

Just as ensemble learning has been proved to be more useful compared to single-model solutions for classification and regression problems, one may expect that cluster ensembles will improve the quality of results as compared to a single clustering solution. It has been shown that using cluster ensembles leads to more accurate results on average as the ensemble approach takes into account the biases of individual solutions [41, 33].

(b) **Robust Clustering**

It is well known that the popular clustering algorithms often fail spectacularly for certain datasets that do not match well with the modeling assumptions [35]. A cluster ensemble approach can provide a "meta" clustering model that is much more robust in the sense of being able to provide good results across a very wide range of datasets. As an example, by using an ensemble that includes approaches such as $k$-means, SOM and DBSCAN that are typically better suited to low-dimensional metric spaces, as well as base clusterers designed for high dimensional sparse spaces (spherical $k$-means, Jaccard based graph clustering, etc.), one can perform well across a wide range of data dimensionality [59]. Authors in [56] present several empirical results on the robustness of the results in document clustering by using feature diversity and consensus clustering.

(c) **Model Selection**

Cluster ensembles provide a novel approach to the model selection problem by considering the match across the base solutions to determine the final number of clusters to be obtained [27].

(d) **Knowledge Reuse**

In certain applications, domain knowledge in the form of a variety of clusterings of the objects under consideration may already exist due to past projects. A consensus

solution can integrate such information to get a more consolidated clustering. Several examples are provided in [59], where such scenarios formed the main motivation for developing a consensus clustering methodology. As another example, a categorization of web pages based on text analysis can be enhanced by using the knowledge of topical document hierarchies available from Yahoo! or DMOZ.

(e) **Multi-view Clustering**

Often the objects to be clustered have multiple aspects or "views", and base clusterings may be built on distinct views that involve non-identical sets of features or subsets of data points. In marketing applications for example, customers may be segmented based on their needs, psychographic or demographic profiles, attitudes etc. Different views can also be obtained by considering qualitatively different distance measures, an aspect that was exploited in clustering multifaceted proteins to multiple functional groups in [6]. Consensus clustering can be effectively used to combine all such clusterings into a single consolidated partition. Strehl & Ghosh [59] illustrated empirically the utility of cluster ensembles in two orthogonal scenarios,

- Feature Distributed Clustering (FDC): different base clusterings are built by selecting different subsets of the features but utilizing all the data points.
- Object Distributed Clustering (ODC): base clusterings are constructed by selecting different subsets of the data points but utilizing all the features.

Fern & Brodley [20] proposed multiple random projections of the data onto subspaces followed by clustering of project data and subsequent aggregation of clustering results as a strategy for high dimensional clustering. They showed that such method performs better than applying PCA and clustering in the reduced feature space.

(f) **Distributed Computing**

In certain situations, data is inherently distributed and it is not possible to first collect the entire data at a central site due to privacy/ownership issues or computational, bandwidth and storage costs [49]. An ensemble can be used in situations where each clusterer has access to only a subset of the features of each object, as well as where each clusterer has access to only a subset of the objects [27], [59].

The problem of combining multiple clusterings can be viewed as a special case of the more general problem of comparison and consensus of data "classifications", studied in the pattern recognition and related application communities in the 70's and 80's. In this literature, "classification" was used in a broad sense to include clusterings, unrooted trees, graphs, etc, and problem-specific formulations were made (see [50] for a broad, more conceptual coverage). For example, in the building of phylogenetic trees, it is important to get a strict consensus solution, wherein two objects occur in the same consensus partition if and only if they occur together in all individual clusterings [15], typically resulting in a consensus solution at a much coarser resolution than the individual solutions. A quick overview with pointers to such literature is given by Ayad and Kamel [7].

This chapter is organized as follows. In Section 1.2, we formulate the cluster ensemble problem. In Section 1.3, different measures for comparing a pair of clustering solutions are introduced. Details of different cluster ensembles algorithms are presented in Section 1.4. In Section 1.5, a summary of recent works on combining classifier and cluster ensembles is illustrated. Finally, the applications of cluster ensembles are provided in Section 1.6.

## 1.2 The Cluster Ensemble Problem

We denote a vector by a bold faced letter and a scalar variable or a set in normal font. We start by considering $r$ base clusterings of a data set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{n}$ with the $q^{th}$ clustering containing $k^{(q)}$ clusters. The most straightforward representation of the $q^{th}$ clustering is $\lambda^{(q)} = \{\mathcal{C}_\ell | \ell = 1, 2, ..., k^{(q)} \text{ and } \mathcal{C}_\ell \subseteq \mathcal{X}\}$. Here, each clustering is denoted by a collection of subsets (not necessarily disjoint) of the original dataset. For hard partitional clustering (clustering where each object is assigned to a single cluster only), the $q^{th}$ clustering can alternatively be represented by a label vector $\lambda^{(q)} \in \mathbb{Z}_+^n$. In this representation, each object is assigned some cluster label and 0 is used if the corresponding object was not available to that clusterer. The third possible way of representation of an individual clustering is by the binary membership indicator matrix $\mathbf{H}^q \in \{0,1\}^{1 \times k^{(q)}}$ which is defined as $\mathbf{H}^q = \{h_{i\ell}^q | h_{i\ell}^q \in \{0,1\}\ \forall \mathbf{x}_i, \mathcal{C}_\ell, \lambda^{(q)}\}$. For partitional clustering, we additionally have $\sum_{\ell=1}^{k^{(q)}} h_{i\ell}^q = 1\ \forall \mathbf{x}_i \in \mathcal{X}$.

A *consensus function* $\Gamma$ is defined as a function $\mathbb{Z}_+^{n\times r} \to \mathbb{Z}_{++}^n$ mapping a set of clusterings to an integrated clustering $\Gamma : \lambda^{(q)}|q \in \{1, 2, ....., r\} \to \hat{\lambda}$. For conciseness, we shall denote the set of clusterings $\{\lambda^{(q)}\}_{q=1}^r$ that is available to the consensus mechanism by $\Lambda$. Moreover, the results of any hard clustering[2] of $n$ objects can be represented as a binary, symmetric $n \times n$ *co-association matrix*, with an entry being 1 if the corresponding objects are in the same cluster and 0 otherwise. For the $q^{\text{th}}$ base clustering, this matrix is denoted by $S^{(q)}$ and is given by

$$S_{ij}^{(q)} = \begin{cases} 1 & (i,j) \in C_\ell(\lambda^{(q)}) \text{ for some } \ell \in \{1, 2, ..., k^{(q)}\} \\ 0 & \text{otherwise} \end{cases} \tag{1.1}$$

Broadly speaking, there are two main approaches to obtaining a consensus solution and determining its quality. One can postulate a probability model that determines the labeling of the individual solutions, given the true consensus labels, and then solve a maximum likelihood formulation to return the consensus [63, 67]. Alternately, one can directly seek a consensus clustering that agrees the most with the original clusterings. The second approach requires a way of measuring the similarity between two clusterings, for example to evaluate how close the consensus solution is to each base solution. These measuring indices will be discussed in more details in Section 1.3. For now, let $\phi(\lambda^{(a)}, \lambda^{(b)})$ represent a similarity index between two clustering solutions $\lambda^{(a)}$ and $\lambda^{(b)}$. One can express the average normalized similarity measure between a set of $r$ labelings, $\Lambda$, and a single consensus labeling $\hat{\lambda}$, by:

$$\phi(\Lambda, \hat{\lambda}) = \frac{1}{r}\sum_{q=1}^{r} \phi(\lambda^{(q)}, \hat{\lambda}). \tag{1.2}$$

This serves as the objective function in certain cluster ensemble formulations, where the goal is to find the combined clustering $\hat{\lambda}$ with $\hat{k}$ clusters such that $\phi(\Lambda, \hat{\lambda})$ is maximized. It turns out though that this objective is intractable, so heuristic approaches have to be resorted to.

[2]This definition is also valid for overlapping clustering.

Table 1.1: Contingency Table Explaining Similarity Measurement of Clustering Solutions

| | $\mathcal{C}_1{}^{(b)}$ | $\mathcal{C}_2{}^{(b)}$ | $\cdots$ | $\mathcal{C}_{k^{(b)}}{}^{(b)}$ | sum |
|---|---|---|---|---|---|
| $\mathcal{C}_1{}^{(a)}$ | $n_{11}$ | $n_{12}$ | $\cdots$ | $n_{1k^{(b)}}$ | $n_1{}^{(a)}$ |
| $\mathcal{C}_2{}^{(a)}$ | $n_{21}$ | $n_{22}$ | $\cdots$ | $n_{2k^{(b)}}$ | $n_2{}^{(a)}$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $\mathcal{C}_{k^{(a)}}{}^{(a)}$ | $n_{k^{(a)}1}$ | $n_{k^{(a)}2}$ | $\cdots$ | $n_{k^{(a)}k^{(b)}}$ | $n_{k^{(a)}}{}^{(a)}$ |
| sum | $n_1{}^{(b)}$ | $n_2{}^{(b)}$ | $\cdots$ | $n_{k^{(b)}}{}^{(b)}$ | $n$ |

## 1.3 Measuring Similarity Between Clustering Solutions

Since no ground truth is available for clustering problems, cluster ensemble algorithms instead aim to maximize some similarity measure between the consensus clustering and each of the base clustering solutions. Two of the most desirable properties of such similarity measures are:

- The index should be normalized for easy interpretation and comparison across solutions with varying number of clusters.
- The expected value of the index between pairs of independent clusterings should be constant (and preferably zero indicating no similarity). The utility of such a property will be clarified later in this section.

Below, we present different indices and show how these indices are modified to achieve zero expected value of the same. The general rule for any "adjustment for chance" is as follows:

$$\text{adjusted index} = \frac{\text{index - Expected-index}}{\text{Max-index - Expected-index}}, \tag{1.3}$$

where Expected-index is calculated according to a permutation model [42]. In such a model, cluster labels are assumed to be generated randomly subject to the constraints of a fixed number of clusters and a fixed number of points in each cluster. Max-index is the maximum value the index can take.

Now let us discuss two key approaches for measuring the similarity of two clustering solutions [53]. The first approach is based on counting the number of pairs in agreement or in disagreement in two clustering solutions. The second approach uses concepts from information theory.

(a) **Pair Counting Based Measures:** Measures of this type are built on counting the number of pairs of points on which two candidate clustering solutions agree or disagree. More formally, suppose we have two candidate clusterings $\lambda^{(a)} = \{\mathcal{C}_h^{(a)} | h = 1, 2, ..., k^{(a)}\}$ and $\lambda^{(b)} = \{\mathcal{C}_\ell{}^{(b)} | \ell = 1, 2, ..., k^{(b)}\}$. Let $n_h^{(a)}$ be the number of objects in cluster $C_h^{(a)}$ and $n_l^{(b)}$ be the number of objects in cluster $C_l^{(b)}$. Table 1.3 is a contingency table that shows the overlap between different clusters of these clusterings, where $n_{h\ell} = |\mathcal{C}_h^{(a)} \cap C_\ell^{(b)}|$. The most well-known index of this class is the Rand Index (**RI**) which is a normalized measure of number of agreements between two candidate solutions and is defined as:

$$\phi^{(RI)}(\lambda^{(a)}, \lambda^{(b)}) = \sum_{h\ell} \begin{pmatrix} n_{h\ell}^{(a)} \\ 2 \end{pmatrix} / \frac{1}{2}(S_a + S_b). \tag{1.4}$$

Expected value of **RI**, however, is not constant. This led Hubert & Arabie [34] to propose the Adjusted Rand Index (**ARI**) to correct for a zero baseline. **ARI**, according to the general strategy of correcting for chance, is defined as follows:

$$\phi^{(ARI)}(\lambda^{(a)}, \lambda^{(b)}) = \frac{\sum_{h\ell} \begin{pmatrix} n_{h\ell} \\ 2 \end{pmatrix} - S_a S_b / \begin{pmatrix} n \\ 2 \end{pmatrix}}{\frac{1}{2}(S_a + S_b) - S_a S_b / \begin{pmatrix} n \\ 2 \end{pmatrix}} \tag{1.5}$$

where $S_a = \sum_h \begin{pmatrix} n_h^{(a)} \\ 2 \end{pmatrix}$ and $S_b = \sum_\ell \begin{pmatrix} n_\ell^{(b)} \\ 2 \end{pmatrix}$. The second term in both numerator and denominator adjusts for the expected number of overlaps that will occur "by chance". Values of **RI** lie between 0 and 1. However, values of **ARI** can be negative which is of no practical interest. **ARI** is 1 when the two candidate clustering solutions match exactly and is 0 when the index value equals its expected value.

In [5], the authors mention as many as 22 different indices of this class. Subsequent work [69] shows that after correction for chance, some of these indices become equivalent. However, **ARI** remains the most popular index of this class. Very recently, a probabilistic version of Rand Index (**PRI** – [13]) has been proposed in which the agreements and disagreements of the pairs of data points are weighted according to their occurrence by chance.

(b) **Information Theoretic Based Measures:**

There are numerous information theory based measures in the literature [53]. We discuss only two of them here and show how they can be corrected for occurrence by chance.

(a) **Normalized Mutual Information (NMI)**

Strehl & Ghosh [59] proposed **NMI** to measure the similarity between two candidate clusterings. The entropy associated with clustering $\lambda^{(a)}$ is $H(\lambda^{(a)}) = -\sum_h \frac{n_h^{(a)}}{n} log(\frac{n_h^{(a)}}{n})$ and that with clustering $\lambda^{(b)}$ is $H(\lambda^{(b)}) = -\sum_\ell \frac{n_\ell^{(b)}}{n} log(\frac{n_\ell^{(b)}}{n})$. Similarly, the joint entropy of $\lambda^{(a)}$ and $\lambda^{(b)}$ is defined as, $H(\lambda^{(a)}, \lambda^{(b)}) = -\sum_{h,\ell} \frac{n_{h\ell}}{n} log(\frac{n_{h\ell}}{n})$. Now, the **NMI** between $\lambda^{(a)}$and $\lambda^{(b)}$ is defined as:

$$\phi^{(NMI)}(\lambda^{(a)}, \lambda^{(b)}) = \frac{I(\lambda^{(a)}, \lambda^{(b)})}{\sqrt{H(\lambda^{(a)})H(\lambda^{(b)})}} \tag{1.6}$$

Here, $I(\lambda^{(a)}, \lambda^{(b)}) = H(\lambda^{(a)}) + H(\lambda^{(b)}) - H(\lambda^{(a)}, \lambda^{(b)})$ is the mutual information between two clusterings $\lambda^{(a)}$ and $\lambda^{(b)}$ ([8]), which is normalized by the geometric mean of $H(\lambda^{(a)})$ and $H(\lambda^{(b)})$ to compute the **NMI**. It should be noted that $I(\lambda^{(a)}, \lambda^{(b)})$ is non-negative and has no upper bound. $\phi^{(NMI)}(\lambda^{(a)}, \lambda^{(b)})$, on the other hand, lies between 0 and 1 and is suitable for easier interpretation and comparisons.

(b) **Variation of Information (VI)**

**VI** is another information theoretic *distance* measure proposed for cluster validation [46, 47], and is defined as follows:

$$\phi^{(VI)}(\lambda^{(a)}, \lambda^{(b)}) = H(\lambda^{(a)}) + H(\lambda^{(b)}) - 2I(\lambda^{(a)}, \lambda^{(b)}) \tag{1.7}$$

It turns out that **VI** is a metric. But its original definition is not consistent if data sets of different sizes and clusterings with different number of clusters are

considered. Therefore, several normalized versions of **VI** have been proposed. The one proposed by [39] takes the following form:

$$\phi^{(\mathrm{NVI}_1)}(\lambda^{(a)}, \lambda^{(b)}) = 1 - \frac{I(\lambda^{(a)}, \lambda^{(b)})}{\max\{H(\lambda^{(a)}), H(\lambda^{(b)})\}}, \tag{1.8}$$

which again is a metric. However, the one proposed by Wu *et. al.* [71] is not a metric and takes the following form:

$$\phi^{(\mathrm{NVI}_2)}(\lambda^{(a)}, \lambda^{(b)}) = 1 - \frac{2I(\lambda^{(a)}, \lambda^{(b)})}{H(\lambda^{(a)}) + H(\lambda^{(b)})}. \tag{1.9}$$

Both of these measures lie between 0 and 1. Note that one could also define a distance measure from **NMI** just by taking its 1-complement. However, such distance measure is not a metric.

Vinh *et. al.* [53] empirically showed that like **RI**, the information theoretic based indices do not attain any constant baseline. The problem is more severe when the number of clusters is comparable to number of data points. With increase in number of clusters, the expected values of the indices either increase (for **NMI**) or decrease (for **VI**). Therefore, if one needs to make a decision based on the observed values of the indices, on average, a clustering solution with more (or less for **VI**) number of clusters will unjustifiably be preferred.

Therefore, following the generalized suggestion for correction by chance given in Eq. 1.3, Vinh *et. al.* [53] proposed to correct these measures with the expected value of the mutual information of the solutions and empirically showed the invariance of the modified measures w.r.t the number of clusters. The expression for the expected value of the mutual information under the permutation model is given by:

$$\begin{aligned}\mathbb{E}[I(S_a, S_b)] \;=\; & \sum_{h\ell} \sum_{n_{h\ell}=\max\{n_h^{(a)}+n_\ell^{(b)}-n,0\}}^{\min\{n_h^{(a)}+n_\ell^{(b)}\}} \frac{n_{h\ell}}{n}\log\left(\frac{n_{h\ell}n}{n_h^{(a)}n_\ell^{(b)}}\right) \\ & \cdot \frac{n_h^{(a)}!n_\ell^{(b)}!(n-n_h^{(a)})!(n-n_\ell^{(b)})!}{n_{h\ell}!n!(n_h^{(a)}-n_{h\ell})!(n_\ell^{(b)}-n_{h\ell})!(n-n_h^{(a)}-n_\ell^{(b)}+n_{h\ell})!}\end{aligned} \tag{1.10}$$

However, if the ratio of number of data points and the number of clusters is more than 100 for both solutions, empirical studies show that $\mathbb{E}[I(S_a, S_b)]$ is fairly close to zero and hence no adjustment for chance is necessary. Unfortunately, with such adjustment for chance, none of the distances measures is a metric anymore. Therefore, depending on the applications, one needs to make a choice between the metric property and the zero baseline property. We would also like to point out that in a recent paper [48], attempt has been made to understand the geometry of partitionings. In particular, the equivalence among some other distance measures like Hamming distance, mis-classification error distance and, $\chi^2$ distance has been established under fairly mild assumptions, thereby opening a new direction of research where the cluster ensemble algorithms can possibly be designed to directly optimize some of these distance measures.

## 1.4 Cluster Ensemble Algorithms

Cluster ensemble methods are now presented under three categories: i) probabilistic approaches, ii) approaches based on co-association and iii) direct and other heuristic methods.

### 1.4.1 Probabilistic Approaches to Cluster Ensembles

The two basic probabilistic models for solving cluster ensembles are described in this subsection.

#### A Mixture Model for Cluster Ensembles (MMCE)

In a typical mixture model [12] approach to clustering, such as fitting the data using a mixture of Gaussians, there are $\hat{k}$ mixture components, one for each cluster. A component-specific parametric distribution is used to model the distribution of data attributed to a specific component. Such an approach can be applied to form the consensus decision if the number of consensus clusters is specified. This immediately yields the pioneering approach taken

in [63]. We describe it in a bit more detail as this work is essential to build an understanding of later works [67, 68].

In the basic mixture model of cluster ensembles [63], each object $\mathbf{x}_i$ is represented by $\mathbf{y}_i = \Lambda(\mathbf{x}_i)$, i.e, the labels provided by the base clusterings. We assume that there are $\hat{k}$ consensus clusters each of which is indexed by $\hat{\ell}$. Corresponding to each consensus cluster $\hat{\ell}$ and each base clustering $q$, we have a multinomial distribution $\boldsymbol{\beta}_{\hat{\ell}}^{(q)}$ of dimension $k^{(q)}$. Therefore, a sample from this distribution is a cluster label corresponding to the $q^{\text{th}}$ base clustering. The underlying generative process is assumed as follows:

For $i^{\text{th}}$ data point $\mathbf{x}_i$,

(a) Choose $\mathbf{z}_i = \mathbf{I}_{\hat{\ell}}$ such that $\hat{\ell} \sim$ multinomial($\boldsymbol{\theta}$). Here $\mathbf{I}_{\hat{\ell}}$ is a probability vector of dimension $k^{(q)}$ with only the $\hat{\ell}^{\text{th}}$ component being 1, and $\boldsymbol{\theta}$ is a multinomial distribution of dimension $\hat{k}$.

(b) For the $q^{\text{th}}$ base clustering of $i^{\text{th}}$ data point, choose the base clustering result $y_{iq} = \ell \sim$ multinomial($\boldsymbol{\beta}_{\hat{\ell}}^{(q)}$).

These probabilistic assumptions give rise to a simple maximum log-likelihood problem that can be solved using the Expectation Maximization algorithm. This model also takes care of the missing labels in a natural way.

**Bayesian Cluster Ensembles (BCE)**

A Bayesian version of the multinomial mixture model described above was subsequently proposed by Wang *et al* [67]. As in the simple mixture model, we assume $\hat{k}$ consensus clusters with $\boldsymbol{\beta}_{\hat{\ell}}^{(q)}$ being the multinomial distribution corresponding to each consensus cluster $\hat{\ell}$ and each base clustering $q$. The complete generative process for this model is as follows:

For $i^{\text{th}}$ data point $\mathbf{x}_i$,

(a) Choose $\boldsymbol{\theta}_i \sim$ Dirichlet($\boldsymbol{\alpha}$) where $\boldsymbol{\theta}_i$ is a multinomial distribution with dimension $\hat{k}$.

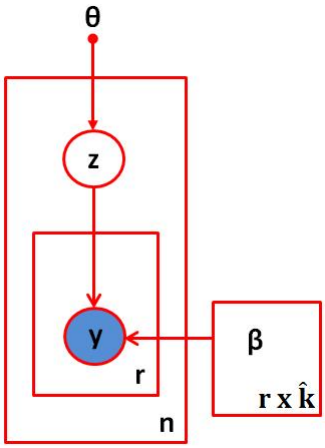


Figure 1.1: Graphical Model for **MMCE**

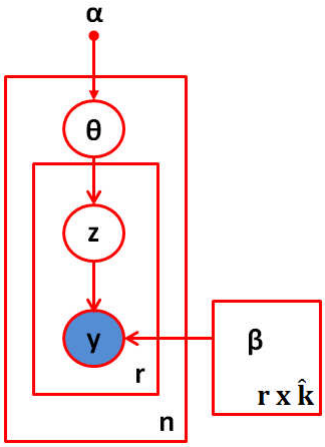


Figure 1.2: Graphical Model for **BCE**

(b) For the $q^{\text{th}}$ base clustering:

(a) Choose $\mathbf{z}_{iq} = \mathbf{I}_{\hat{\ell}}$ such that $\hat{\ell} \sim$ multinomial$(\boldsymbol{\theta}_i)$. $\mathbf{I}_{\hat{\ell}}$ is a probability vector of dimension $\hat{k}$ with only $\hat{\ell}^{\text{th}}$ component being 1.

(b) Choose the base clustering result $y_{iq} = \ell \sim$ multinomial$(\boldsymbol{\beta}_{\hat{\ell}}^{(q)})$.

So, given the model parameters $(\boldsymbol{\alpha}, \beta = \{\boldsymbol{\beta}_{\hat{\ell}}^{(q)}\})$, the joint distribution of latent and observed variables $\{\mathbf{y}_i, \mathbf{z}_i, \boldsymbol{\theta}_i\}$ is given by,

$$p(\mathbf{y}_i, \mathbf{z}_i, \boldsymbol{\theta}_i | \boldsymbol{\alpha}, \beta) = p(\boldsymbol{\theta}_i | \boldsymbol{\alpha}) \prod_{q=1, \exists y_{iq}}^{r} p(\mathbf{z}_{iq} = \mathbf{I}_{\hat{\ell}} | \boldsymbol{\theta}_i) p(y_{iq} | \boldsymbol{\beta}_{\hat{\ell}}^{(q)}) \tag{1.11}$$

where $\exists y_{iq}$ implies that there exists a $q^{\text{th}}$ base clustering result for $\mathbf{y}_i$. The marginals $p(\mathbf{y}_i | \boldsymbol{\alpha}, \beta)$ can further be calculated by integrating over the hidden variables $\{\mathbf{z}_i, \boldsymbol{\theta}_i\}$. The authors used variational EM and Gibb's sampling for inference and parameter estimation. The graphical model corresponding to this Bayesian version is given in figure 1.2. To highlight the difference between Bayesian cluster ensembles and the mixture model for cluster ensembles, the graphical model corresponding to the latter is also shown alongside in figure 1.1.

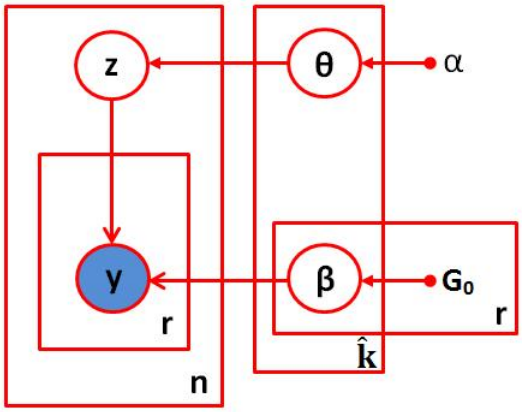


Figure 1.3: Graphical Model for **NPBCE**

**Non-Parametric Bayesian Cluster Ensembles (NPBCE)**

Recently, a non-parametric version of Bayesian cluster ensemble (**NPBCE**) has been proposed in [68] which allows the number of consensus clusters to adapt with data. The stick-breaking construction of the generative process of this model is described below. The authors use a truncated stick breaking construction of the Dirichlet process with truncation enforced at $\hat{k}$. If $\hat{k}$ is made sufficiently large, the resulting Dirichlet Process (Truncated) closely approximates a Dirichlet Process.

(a) Generate $v_{\hat{\ell}} \sim \text{Beta}(1, \alpha)\ \forall \hat{\ell} \in \{1, 2, \cdots, \hat{k}\}$. Let $\theta_{\hat{\ell}} = v_{\hat{\ell}} \prod_{j=1}^{\hat{\ell}-1}(1 - v_j)$.

(b) For each base clustering (indexed by $q$), generate $\boldsymbol{\beta}_{\hat{\ell}}^{(q)} \sim G_0^{(q)}\ \forall \hat{\ell} \in \{1, 2, \cdots, \hat{k}\}$ where $G_0^{(q)}$ is a symmetric Dirichlet distribution of dimension $k^{(q)}$.

(c) For $i^{\text{th}}$ data point $\mathbf{x}_i$, generate $\mathbf{z}_i \sim \text{multinomial}(\boldsymbol{\theta})$. $\mathbf{z}_i$ is an indicator vector of dimension $\hat{k}$ with only one component being unity and others being zero.

(d) For the $q^{\text{th}}$ base clustering of $i^{\text{th}}$ data point, generate the base clustering result $y_{iq} = \ell \sim \text{multinomial}(\boldsymbol{\beta}_{\hat{\ell}}^{(q)})$.

One should note that **NPBCE** does not allow multiple base clustering solutions of a given data point to be generated from more than one consensus cluster. Therefore, the model is more restrictive compared to **BCE** and is really a non-parametric version of **MMCE**. The

graphical model shown in Fig. 1.3 illustrates this difference more clearly. It should be noted that although all of the generative models presented above were used only with hard partitional clustering, they could be used for overlapping clustering as well.

### 1.4.2 Pairwise Similarity Based Approaches

In pairwise similarity based approaches, one takes the weighted average of all $r$ co-association matrices to form an *ensemble co-association matrix* $\mathbf{S}$ which is given as follows,

$$\mathbf{S} = \frac{1}{r} \sum_{q=1}^{r} w_q \mathbf{S}^{(q)}. \tag{1.12}$$

Here $w_q$ specifies the weight assigned to the $q^{\text{th}}$ base clustering. This ensemble co-association matrix captures the fraction of times a pair of data points is placed in the same cluster across the $r$ base clusterings. The matrix can now be viewed as a similarity matrix (with a corresponding similarity graph) to be used by the consensus mechanism for creating the consensus clusters. This matrix is different from the similarity matrix $\hat{\mathbf{S}}$ that we obtain from the consensus solution $\hat{\lambda}$. We will explain the difference in detail in section 1.4.2.

Note that the co-association matrix size is itself quadratic in $n$, which thus forms a lower bound on computational complexity as well as memory requirements, inherently handicapping such a technique for applications to very large datasets. However, it is independent of the dimensionality of the data.

#### Methods based on Ensemble Co-association Matrix

The Cluster-based Similarity Partitioning Algorithm (CSPA) [59] used METIS [36] to partition the induced consensus similarity graph. METIS was chosen for its scalability and because it tries to enforce comparable sized clusters. This added constraint is desirable in several application domains [60]; however, if the data is actually labeled with imbalanced classes, then it can lower the match between cluster and class labels. Assuming quasi-

linear graph clustering, the worst case complexity for this algorithm is $\mathcal{O}(n^2kr)$. Punera & Ghosh [55] later proposed a soft version of CSPA, i.e. one that works on soft base clusterings. Al-Razgan & Domeniconi [4] proposed an alternative way of obtaining non-binary co-association matrices when given access to the raw data.

The Evidence Accumulation approach [23] obtains individual co-association matrices by random initializations of the $k$-means algorithm, causing some variation in the base cluster solutions. This algorithm is used with a much higher value of $k$ than the range finally desired. The ensemble co-association matrix is then formed, each entry of which signifies the relative co-occurrence of two data points in the same cluster. A minimum spanning tree (**MST**) algorithm (also called the single-link or nearest neighbor hierarchical clustering algorithm) is then applied on the ensemble co-association matrix. This allows one to obtain non-convex shaped clusters. Essentially, this approach assumes the designer has access to the raw data, and the consensus mechanism is used to get a more robust solution than what can be achieved by directly applying MST to the raw data.

A related approach was taken by Monti *et al* [51], where the perturbations in the base clustering were achieved by re-sampling. Any of bootstrapping, data sub-sampling or feature sub-sampling can be used as a re-sampling scheme. If either of the first two options are selected, then it is possible that certain objects will be missing in a given base clustering. Hence when collating the $r$ base co-association matrices, the $(i,j)^{\text{th}}$ entry needs to be divided by the number of solutions that included both objects rather than by a fixed $r$. This work also incorporated a model selection procedure as follows: The consensus co-association matrix is formed multiple times. The number of clusters is kept at $k_i$ for each base clustering during the $i^{\text{th}}$ experiment, but this number is changed from one experiment to another. A measurement termed as *consensus distribution* describes how the elements of a consensus matrix are distributed within the 0-1 range. The extent to which the consensus matrix is skewed towards a binary matrix denotes how good the base clusterings match one another. This enables one to choose the most appropriate number of consensus clusters $\hat{k}$. Once $\hat{k}$ is chosen, the corresponding ensemble co-association matrix is fed to a hierarchical clustering algorithm with average linkage. Agglomeration of clusters is stopped when $\hat{k}$ branches are left.

The Iterative Pairwise Consensus (IPC) Algorithm [52] essentially applies model-based $k$-means [75] to the ensemble co-association matrix $S$. The consensus clustering solution $\hat{\lambda} = \{\mathcal{C}_\ell\}_{\ell=1}^{\hat{k}}$ is initialized to some solution, after which a re-assignment of points is carried out based on the current configuration of $\hat{\lambda}$. The point $\mathbf{x}_i$ gets assigned to cluster $\mathcal{C}_\ell$, if $\mathbf{x}_i$ has maximum average similarity with the points belonging to cluster $\mathcal{C}_\ell$. Then the consensus solution is updated, and the cycle starts again.

However, both Mirkin [50] and Li *et al* [45] showed that the problem of consensus clustering can be framed in a different way than what has been discussed so far. In these works, the distance $d(\lambda^{(q_1)}, \lambda^{(q_2)})$ between two clusterings $\lambda^{(q_1)}$ and $\lambda^{(q_2)}$ is defined as the number of pairs of objects that are placed in the same cluster in one of $\lambda^{(q_1)}$ or $\lambda^{(q_2)}$ and in different cluster in the other, essentially considering the (unadjusted) Rand Index. Using this definition, the consensus clustering problem is formulated as,

$$\begin{aligned} \arg\min_{\hat{\lambda}} J &= \arg\min_{\hat{\lambda}} \frac{1}{r}\sum_{q=1}^{r} d(\lambda^{(q)}, \hat{\lambda}) \\ &= \arg\min_{\hat{S}} \frac{1}{r}\sum_{q=1}^{r} w_q \sum_{i<j} [S_{ij}^{(q)} - \hat{S}_{ij}]^2 \end{aligned} \tag{1.13}$$

Mirkin ([50], section 5.3.4, p. 260) further proved that the consensus clustering according to criterion (1.18) is equivalent to clustering over the ensemble co-association matrix by subtracting a "soft" and "uniform" threshold from each of the different consensus clusters. This soft threshold, in fact, serves as a tool to balance cluster sizes in the final clustering. The subtracted threshold has also been used in [62] for consensus clustering of gene-expression data.

In [66], consensus clustering result is obtained by minimizing a weighted sum of the Bregman divergence [9] between the consensus partition and the input partitions wrt their co-association matrices. In addition, the authors also show how to generalize their framework in order to incorporate must-link and cannot-link constraints between objects.

Note that the optimization problem in (1.18) is over the domain of $\hat{S}$. The difference between the matrices $S$ and $\hat{S}$ lies in the way the optimization problem is posed. If optimization is performed with cluster labels only (as illustrated in section 1.4.3), there is no guarantee of achieving the optimum value $\hat{S} = S$. However, if we are optimizing over the domain of the co-association matrix we can achieve this optimum value in theory.

**Relating Consensus Clustering to other Optimization Formulations**

The co-association representation of clustering has been used to relate consensus clustering with two other well-known problems.

(a) **Consensus Clustering as Non-Negative Matrix Factorization (NNMF)**

Li *et al* ([45], [44]), using the same objective function as mentioned in (1.18), showed that the problem of consensus clustering can be reduced to an NNMF problem. Assuming $U_{ij} = \hat{S}_{ij}$ to be a solution to this optimization problem. we can rewrite (1.18) as,

$$\arg\min_{U} \sum_{i,j=1}^{n} (S_{ij} - U_{ij})^2 = \arg\min_{U} \|S - U\|_F^2 \tag{1.14}$$

where the matrix norm is the Frobenius norm. This problem formulation is similar to the NNMF formulation [43] and can be solved using an iterative update procedure. In [29], the cost function $J$ used in equation (1.18) was further modified via normalization to make it consistent with data sets with different number of data points ($n$) and different number of base clusterings ($r$).

(b) **Consensus Clustering as Correlation Clustering**

Gionis *et al* [28] showed that a certain formulation of consensus clustering is a special case of correlation clustering. Suppose we have a data set $\mathcal{X}$ and some kind of dissimilarity measurement (distance) between every pair of points in $\mathcal{X}$. This dissimilarity measure is denoted by $d_{ij} \in [0, 1] \, \forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$. The objective of correlation clustering

[11] is to find a partition $\hat{\lambda}$ such that:

$$\hat{\lambda} = \arg\min_{\lambda} \; d(\lambda)$$

$$= \arg\min_{\lambda} \left[ \sum_{(i,j) : \lambda(\mathbf{x}_i) = \lambda(\mathbf{x}_j)} d_{ij} + \sum_{(i,j) : \lambda(\mathbf{x}_i) \neq \lambda(\mathbf{x}_j)} (1 - d_{ij}) \right] \tag{1.15}$$

In the above equation, $\lambda(\mathbf{x}_i)$ is the cluster label imposed by $\lambda$ on $\mathbf{x}_i$. The co-association view of the cluster ensemble problem reduces to correlation clustering if the distance $d_{ij}$ is defined as $d_{ij} = \frac{1}{r}|\{\lambda^{(q)} : \lambda^{(q)}(\mathbf{x}_i) \neq \lambda^{(q)}(\mathbf{x}_j)\}| \; \forall \; i, j$.

### 1.4.3 Direct Approaches using Cluster Labels

Several consensus mechanisms take only the cluster labels provided by the base clusterings as input, and try to optimize an objective function such as (1.16), without computing the co-association matrix.

#### Graph Partitioning

In addition to CSPA, Strehl & Ghosh [59] proposed two direct approaches to cluster ensembles: Hyper Graph Partitioning Algorithm (HGPA) which clusters the objects based on their cluster memberships, and Meta Clustering Algorithm (MCLA), which groups the clusters based on which objects are contained in them. HGPA considers a graph with each object being a vertex. A cluster in any base clustering is represented by a hyper-edge connecting the member vertices. The hyper-graph clustering package HMETIS (Karypis *et al* [37]) was used as it gives quality clusterings and is very scalable. As with CSPA, employing a graph clustering algorithm adds a constraint that favors clusterings of comparable size. Though HGPA is fast with a worst case complexity of $\mathcal{O}(nkr)$, it suffers from an additional problem: if all members of a base cluster are not assigned the same cluster in the consensus solution, the corresponding hyper-edge is broken and incurs a constant penalty; however it

cannot distinguish between a situation where only one object was clustered differently and one where several objects were allocated to other groups. Due to this issue, HGPA is often not competitive in terms of cluster quality.

MCLA first forms a meta-graph with a vertex for each base cluster. The edge weights of this graph are proportional to the similarity between vertices, computed using the binary Jaccard measure (number of elements in common divided by the total number of distinct elements). Since the base clusterings are partitional, this results in an r-partite graph. The meta-graph is then partitioned into $k$ balanced meta-clusters. Each meta-cluster, therefore, contains approximately $r$ vertices. Finally, each object is assigned to its most closely associated meta-cluster. Ties are broken randomly. The worst case complexity is $\mathcal{O}(nk^2r^2)$.

Noting that CSPA and MCLA consider either the similarity of objects or similarity of clusters only, a Hybrid Bipartite Graph Formulation (HBGF) was proposed in [18]. A bipartite graph models both data points and clusters as vertices, wherein an edge exists only between a cluster vertex and a object vertex if the latter is a member of the former. Either METIS or other multi-way spectral clustering methods are used to partition this bipartite graph. The corresponding soft versions of CSPA, MCLA and HBGF have also been developed by Punera & Ghosh [55]. It should be noted that all of CSPA, MCLA and HGPA were compared with one other using the NMI measure in [59].

### Cumulative Voting

The concept of cumulative voting was first introduced in [17] where the authors used bagging to improve the accuracy of clustering procedure. Once clustering is done on a bootstrapped sample, the cluster correspondence problem is solved using iterative re-labeling via Hungarian algorithm. Clustering on each bootstrapped sample gives some votes corresponding to each data point and cluster label pair which, in aggregate, decides the final cluster assignment.

A similar approach was adopted in [7]. Each base clustering in this contribution is thought of as providing a soft or probabilistic vote on which clusters in the consensus solution its data points should belong to. These votes are then gathered across the base solutions and thresh-

olded to determine the membership of each object to the consensus clusters. Again, this requires a mapping function from the base clusterings to a stochastic one. An information-theoretic criterion based on the information bottleneck principle was used in [7] for this purpose. The mean of all the stochastic clusterings then yields the consensus partition. This approach is able to cater to a range of "$k$" in the base clusterings, is fast as it avoids the quadratic time/space complexity of forming a co-association matrix, and has shown good empirical results as well. Noting that the information bottleneck solutions can be obtained as a special case of Bregman clustering [9], it should be possible to recast this approach as a probabilistic one.

A variety of heuristic search procedures have also been suggested to hunt for a suitable consensus solution. These include a genetic algorithm formulation [74] and one using a multi-ant colony [72]. These approaches tend to be computationally expensive and the lack of extensive comparisons with the methods covered in this article currently make it difficult to assess their quality. Also, one can use several heuristics suggested in [19] to select only a few clustering solutions from a large ensemble.

## 1.5 Combination of Classifier and Clustering Ensembles

Based on the success of classifier and cluster ensembles, efforts have been made recently to combine the strength of both types of ensembles [1, 25]. Unsupervised models can provide a variety of supplementary constraints which can be useful for improving the generalization capability of a classifier (or a classifier ensemble), specially when labeled data is scarce. Also, they might be useful for designing learning methods that are aware of the possible differences between training and target distributions, thus being particularly interesting for applications in which concept drift might take place [3, 24]. This section focuses on one such algorithm [1, 3] named C$^3$E (from **C**ombination of **C**lassification and **C**lustering **E**nsembles).

The framework described in [1, 3] is depicted in Fig. 1.4. A set of classifiers, previously induced from a training dataset, is applied on a new target data $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$, thereby generating a set of average class probability estimates $\{\boldsymbol{\pi}_i\}_{i=1}^n$ on the instances in the target

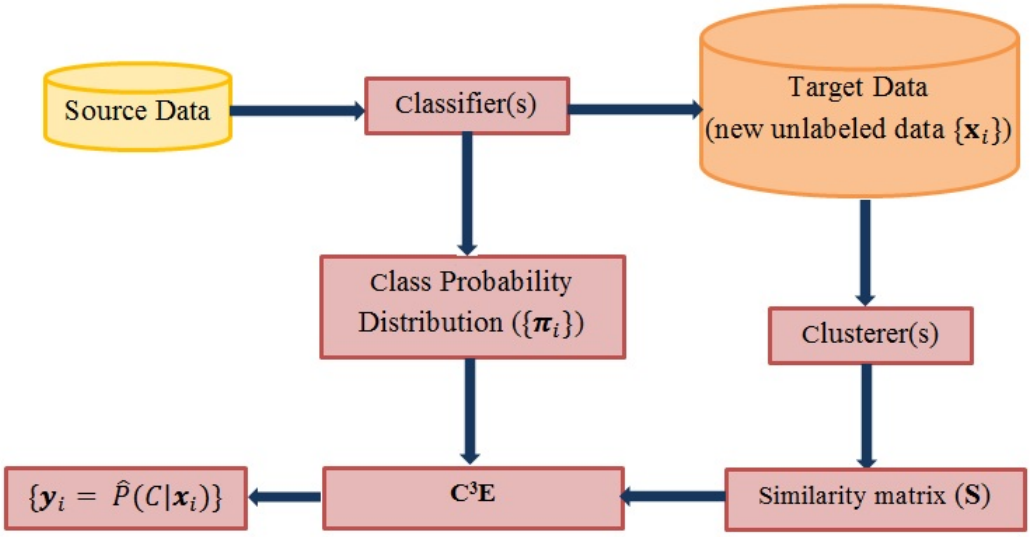


Figure 1.4: Overview of **C³E**.

data. Suppose there are $k$ classes denoted by $C = \{C_\ell\}_{\ell=1}^{k}$. A cluster ensemble is further applied on the target data and a similarity matrix $\mathbf{S}$ is obtained from the ensemble outputs. Each entry of this similarity matrix corresponds to the relative co-occurrence of two instances in the same cluster [59]. The prediction from the classifier ensemble ($\{\boldsymbol{\pi}_i\}_{i=1}^{n}$) is then refined with the help of the similarity matrix obtained from cluster ensemble. The refined class assignment estimate can be represented by a set of vectors $\{\mathbf{y}_i\}_{i=1}^{n}$ with $\mathbf{y}_i \in \mathcal{S} \subseteq \mathbb{R}^k \ \forall i$, and $\mathbf{y}_i \propto \hat{P}(C \mid \mathbf{x}_i)$ (estimated posterior class probability assignment).

The problem of combining classifiers and clusterers is then posed as an optimization problem whose objective is to minimize $J$ in (1.16) with respect to the set of vectors $\{\mathbf{y}_i\}_{i=1}^{n}$:

$$J = \sum_{i \in \mathcal{X}} d_\phi(\boldsymbol{\pi}_i, \mathbf{y}_i) + \alpha \sum_{(i,j) \in \mathcal{X}} s_{ij} d_\phi(\mathbf{y}_i, \mathbf{y}_j) \tag{1.16}$$

The term $d_\phi(.,.)$ refers to a Bregman divergence [10]. Informally, the first term in Eq. (1.16) captures dissimilarities between the class probabilities provided by the ensemble of classifiers and the output vectors $\{\mathbf{y}_i\}_{i=1}^{n}$. The second term encodes the cumulative weighted dissimilarity between all possible pairs $(\mathbf{y}_i, \mathbf{y}_j)$. The weights to these pairs are assigned in proportion to the similarity values $s_{ij} \in [0, 1]$ of matrix $\mathbf{S}$. The coefficient $\alpha \in \mathbb{R}_+$ controls the

relative importance of classifier and cluster ensembles. Therefore, minimizing the objective function over $\{\mathbf{y}_i\}_{i=1}^n$ involves combining the evidence provided by the ensembles in order to build a more consolidated classification.

All Bregman divergences have the remarkable property that the single best (in terms of minimizing the net loss) representative of a set of vectors, is simply the expectation of this set (!) provided the divergence is computed with this representative as the second argument of $d_\phi(\cdot,\cdot)$ — see Theorem 1 in [1] (originally from [10]) for further reference. Unfortunately, this simple form of the optimal solution is not valid if the variable to be optimized occurs as the first argument. In that case, however, one can work in the (Legendre) dual space, where the optimal solution has a simple form. Re-examination of Eq. (1.16) reveals that the $\mathbf{y}_i$'s to be minimized over occur both as first and second arguments of a Bregman divergence. Hence optimization over $\{\mathbf{y}_i\}_{i=1}^n$ is not available in closed form. This problem is circumvented by creating two copies for each $\mathbf{y}_i$ — the left copy, $\mathbf{y}_i^{(l)}$, and the right copy, $\mathbf{y}_i^{(r)}$. The left (right) copies are used whenever the variables are encountered in the first (second) argument of the Bregman divergences. In what follows, it will be clear that the right and left copies are updated iteratively, and an additional soft constraint is used to ensure that the two copies of a variable remain "close enough" during the updates. With this modification, the following objective is minimized:

$$J(\mathbf{y}^{(l)},\mathbf{y}^{(r)}) = \left[\sum_{i=1}^{n} d_\phi(\boldsymbol{\pi}_i,\mathbf{y}_i^{(r)}) + \alpha\sum_{i,j=1}^{n} s_{ij} d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_j^{(r)}) + \lambda\sum_{i=1}^{n} d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_i^{(r)})\right], \tag{1.17}$$

where, $\mathbf{y}^{(l)} = \left(\mathbf{y}_i^{(l)}\right)_{i=1}^n \in \mathcal{S}^n$ and $\mathbf{y}^{(r)} = \left(\mathbf{y}_i^{(r)}\right)_{i=1}^n \in \mathcal{S}^n$.

To solve the optimization problem in an efficient way, first $\{\mathbf{y}_i^{(l)}\}_{i=1}^n$ and $\{\mathbf{y}_i^{(r)}\}_{i=1}^n \setminus \{\mathbf{y}_j^{(r)}\}$ are kept fixed, and minimize the objective w.r.t. $\mathbf{y}_j^{(r)}$ only. The associated optimization problem can, therefore, be written as:

$$\min_{\mathbf{y}_j^{(r)}}\left[d_\phi(\boldsymbol{\pi}_j^{(r)},\mathbf{y}_j^{(r)}) + \alpha\sum_{i^{(l)}\in\mathcal{X}} s_{i^{(l)}j^{(r)}} d_\phi(\mathbf{y}_i^{(l)},\mathbf{y}_j^{(r)}) + \lambda_j^{(r)} d_\phi(\mathbf{y}_j^{(l)},\mathbf{y}_j^{(r)})\right], \tag{1.18}$$

where $\lambda_j^{(r)}$ is the corresponding penalty parameter that is used to keep $\mathbf{y}_j^{(r)}$ and $\mathbf{y}_j^{(l)}$ close to

each other.

From the results of Corollary 1 in [1], the unique minimizer of the optimization problem in (1.18) is obtained as:

$$\mathbf{y}_j^{(r)*} = \frac{\boldsymbol{\pi}_j^{(r)} + \gamma_j^{(r)} \sum\limits_{i^{(l)} \in \mathcal{X}} \delta_{i^{(l)}j^{(r)}} \mathbf{y}_i^{(l)} + \lambda_j^{(r)} \mathbf{y}_j^{(l)}}{1 + \gamma_j^{(r)} + \lambda_j^{(r)}}, \tag{1.19}$$

where $\gamma_j^{(r)} = \alpha \sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)}j^{(r)}}$ and $\delta_{i^{(l)}j^{(r)}} = s_{i^{(l)}j^{(r)}} / \left[\sum_{i^{(l)} \in \mathcal{X}} s_{i^{(l)}j^{(r)}}\right]$. The same optimization in (1.18) is repeated over all the $\mathbf{y}_j^{(r)}$'s. After the right copies are updated, the objective function is (sequentially) optimized with respect to all the $\mathbf{y}_i^{(l)}$'s. However, one needs to work in the dual space now where one can get a closed form update again for the $\mathbf{y}_i^{(l)}$'s. The alternating optimization w.r.t. the left and right copies leads to a guaranteed convergence for a jointly convex Bregman divergence [2]. Additionally, a linear rate of convergence has been proven for squared loss, KL divergence and I divergence. The authors in [2] show applications of C$^3$E in semi-supervised and transfer learning scenarios using some standard UCI datasets, real word text classification datasets and, remote sensing datasets. The method has been empirically proven to leverage information from cluster ensembles, particularly in the presence of concept drift.

## 1.6 Applications of Consensus Clustering

The motivation for consensus clustering has already been introduced in Section 1.1. Since cluster ensemble improves the quality of clustering solution, it can be used for any cluster analysis problem, *e.g.* image segmentation [70], bioinformatics, document retrieval [30], automatic malware categorization [73], just to name a few. Gionis *et. al.* [28] showed how clustering ensemble algorithms used to improve the robustness of clustering solution, clustering categorical data [31] and heterogeneous data, identifying the correct number of clusters and detecting outliers. Fischer & Joachim [22] showed how re-sampling the data and subsequent aggregation of the clustering solutions from the sampled sets can improve

the quality of clustering solution. Sawtooth Software (`http://www.sawtoothsoftware.com/`) has commercialized some of the algorithms in [59] for applications in marketing. A package consisting of implementations of all the algorithms in [59] is also available on `http://strehl.com/soft.html`. In this section, we briefly discuss two major application domains.

### 1.6.1 Gene Expression Data Analysis

Consensus clustering has been applied to microarray data to improve the quality and robustness of the resulting clusters. A resampling based approach is used by Monti *et. al.* [51], in which the agreement across the results obtained by executing a base clustering algorithm on several perturbations of the original dataset is used to obtain the final clustering. Swift *et. al.* [61] use a variety of clustering algorithms on the same dataset to generate different base clustering results and try to find clusters that are consistent across all the base results using simulated annealing. In [21] consensus clustering problem is treated as a median partition problem, where the aim is to find a partitioning of the data points that minimizes the distance to all the other partitionings. The authors propose greedy heuristic solutions to find a local optimum. Additionally, Deodhar & Ghosh [16] used consensus clustering to find overlapping clusters in micro-array data. In this work, two different techniques are used to generate the consensus clustering solution from the candidate solutions. The first one is **MCLA** with some adjustable threshold and the second one is soft kernelized $k$-means that works on ensemble co-association matrix. In [14], gene expression time series data is clustered at different time intervals and the solutions from different time stamps are merged in a single solution using graph partitioning of the ensemble co-association matrix.

### 1.6.2 Image Segmentation

Though there exists several image segmentation algorithms, depending on the application data, some perform better than the others and it is almost impossible to know beforehand which one should be used. The authors in [58] used the cluster ensemble formulation to aggregate the results of multiple segmentation algorithms like (a) Normalized Cuts, (b) En-

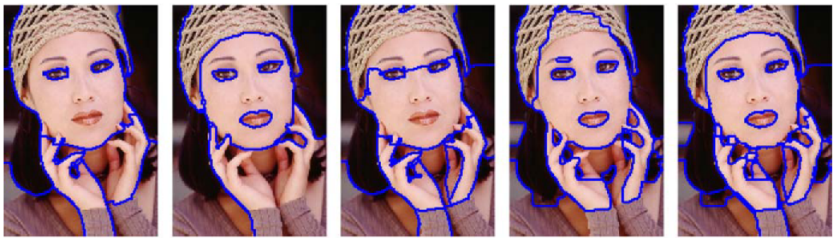

Figure 1.5: Segmentation Result 1 – from [58]

ergy Minimization by Graph Cuts, and (c) Curve Evolution to generate image segmentation. However, they found that the ensemble segmentation outperforms any individual segmentation algorithm. One of such results is shown in Fig. 1.5. The first result corresponds to Normalized Cuts, the second one is from Graph Cuts, the third and fourth ones are from Curve Evolution, and the last one is due to the ensemble segmentation.

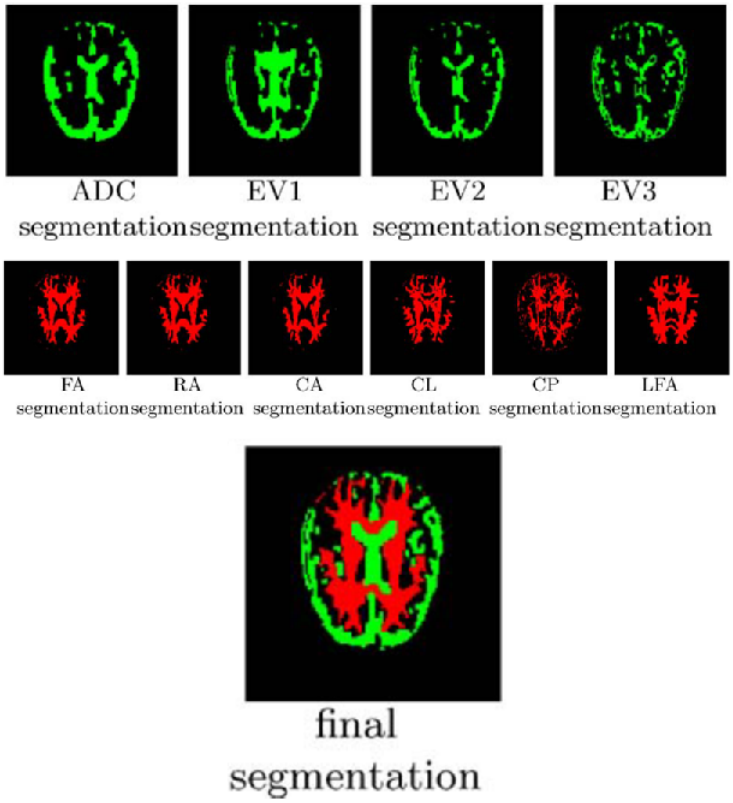


Figure 1.6: Segmentation Result 2 – from [58]

In another similar application, the same authors showed the utility of ensemble methods for better visualization and interpretation of images obtained from Diffusion Tensor Imaging

(**DTI**) technique. **DTI** images have become popular within neuroimaging because they are useful to infer the underlying structure and organizational pattern in the body (e.g., neuronal pathways in the brain). To simplify the processing of such images, a number of different measures (or channels) are calculated from the diffusion tensor image. Some of these channels are Apparent Diffusion Coefficient (ADC), Fractional Anisotropy (FA), Mean Diffusivity (MD), Planar anisotropy (CP) etc[3]. Segmentations obtained from 10 of such channels of a brain are given in the first two rows of Fig. 1.6. The last row shows the segmentation obtained using the ensemble strategy.

## 1.7 Concluding Remarks

This article first showed that cluster ensembles are beneficial in a wide variety of scenarios. It then provided a framework for understanding many of the approaches taken so far to design such ensembles. Even though there seems to be many different algorithms for this problem, we showed that there are several commonalities among these approaches. The design domain, however, is still quite rich leaving space for more efficient heuristics as well as formulations that place additional domain constraints to yield consensus solutions that are useful and actionable in diverse applications.

## Acknowledgement

This work has been supported by NSF Grants (IIS-0713142 and IIS-1016614), ONR Grant (ATL N00014-11-1-0105) and NHARP.

## References

[1] A. Acharya, E. R. Hruschka, J. Ghosh, and S. Acharyya. C$^3$E: A Framework for Combining Ensembles of Classifiers and Clusterers. In *10th Int. Workshop on MCS*,

[3] Please see [58] for more details about all of the 10 channels.

2011.

[2] A. Acharya, E.R. Hruschka, J. Ghosh, and S. Acharyya. An optimization framework for semi-supervised and transfer learning using multiple classifiers and clusterers. *CoRR*, abs/1206.0994, 2012.

[3] A. Acharya, E.R. Hruschka, J. Ghosh, and S. Acharyya. Transfer learning with cluster ensembles. *JMLR Workshop and Conference Proceedings*, 27:123–132, 2012.

[4] M. Al-Razgan and C. Domeniconi. Weighted cluster ensemble. In *Proceedings of SIAM International Conference on Data Mining*, pages 258–269, 2006.

[5] Ahmed N. Albatineh, Magdalena Niewiadomska-Bugaj, and Daniel Mihalko. On similarity indices and correction for chance agreement. *Journal of Classification*, 23(2):301–313, September 2006.

[6] Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar. An ensemble framework for clustering protein-protein interaction networks. In *In Proc. 15th Annual Intl Conference on Intelligent Systems for Molecular Biology (ISMB*, page 2007, 2007.

[7] Hanan G. Ayad and Mohamed S. Kamel. Cumulative voting consensus method for partitions with variable number of clusters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(1):160–173, 2008.

[8] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Jl. Machine Learning Research (Journal of Machine Learning Research)*, 6:1345–1382, 2005.

[9] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Jl. Machine Learning Research (Journal of Machine Learning Research)*, 6:1705–1749, October 2005.

[10] A. Banerjee, S. Merugu, Inderjit S. Dhillon, and J. Ghosh. Clustering with bregman divergences. *J. Machine Learning Res.*, 6:1705–1749, December 2005.

[11] N. Bansal, A.L. Blum, and S. Chawla. Correlation clustering. In *Proceedings of Foundations of Computer Science*, page 238247, 2002.

[12] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[13] Claudio Carpineto and Giovanni Romano. Consensus clustering based on a new probabilistic rand index with application to subtopic retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(PrePrints), 2012.

[14] Tai-Yu Chiu, Ting-Chieh Hsu, and Jia-Shung Wang. Ap-based consensus clustering for gene expression time series. *Pattern Recognition, International Conference on*, 0:2512–2515, 2010.

[15] W.H.E. Day. Foreword: Comparison and consensus of classifications. *J. Classification*, 3:183–185, 1986.

[16] Meghana Deodhar and Joydeep Ghosh. Consensus clustering for detection of overlapping clusters in microarray data. In *ICDMW '06: Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, pages 104–108, Washington, DC, USA, 2006. IEEE Computer Society.

[17] Sandrine Dudoit and Jane Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9):1090–1099, 2003.

[18] X. Fern and C. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proc. of International Conference on Machine Learning*, pages 281–288, 2004.

[19] Xiaoli Z. Fern and Wei Lin. Cluster ensemble selection. *Stat. Anal. Data Min.*, 1(3):128–141, November 2008.

[20] Fern, Xiaoli Z. and Brodley, Carla E. Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach. In *Proc. 20th International Conference on Machine Learning (International Conference on Machine Learning'03)*, Washington, August 2003.

[21] V. Filkov and S. Skiena. Integrating microarray data by consensus clustering. In *International Journal on Artificial Intelligence Tools (IJAIT).*, pages 4:863–880, 2004.

[22] Bernd Fischer and Joachim M. Buhmann. Bagging for Path-based Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1411–1415, November 2003.

[23] A. Fred and A. K. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):835–850, 2005.

[24] J. Gao, W. Fan, J. Jiang, and J. Han. Knowledge transfer via multiple model local structure mapping. In *Proc. of KDD*, pages 283–291, 2008.

[25] Jing Gao, Feng Liang, Wei Fan, Yizhou Sun, and Jiawei Han. A graph-based consensus maximization approach for combining multiple supervised and unsupervised models.

*IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints), 2011.

[26] J. Ghosh. Multiclassifier systems: Back to the future (invited paper). In F. Roli and J. Kittler, editors, *Multiple Classifier Systems*, pages 1–15. LNCS Vol. 2364, Springer, 2002.

[27] J. Ghosh, A. Strehl, and S. Merugu. A consensus framework for integrating distributed clusterings under limited knowledge sharing. In *Proc. NSF Workshop on Next Generation Data Mining, Baltimore*, pages 99–108, Nov 2002.

[28] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data*, 1(4):109–117, March 2007.

[29] A. Goder and V. Filkov. Consensus clustering algorithms: Comparison and refinement. In *Proceedings of the Tenth Workshop on Algorithm Engineering and Experiments*, pages 109–117, 2008.

[30] Edgar Gonzàlez and Jordi Turmo. Comparing non-parametric ensemble methods for document clustering. In *Proceedings of the 13th international conference on Natural Language and Information Systems: Applications of Natural Language to Information Systems*, NLDB '08, pages 245–256, Berlin, Heidelberg, 2008. Springer-Verlag.

[31] Zengyou He, Xiaofei Xu, and Shengchun Deng. A cluster ensemble method for clustering categorical data. *Information Fusion*, 6(2):143 – 151, 2005.

[32] P. Hore, Lawrence O. Hall, and Dmitry B. Goldgof. A scalable framework for cluster ensembles. *Pattern Recogn.*, 42(5):676–688, 2009.

[33] Xiaohua Hu and Illhoi Yoo. Cluster ensemble and its applications in gene expression analysis. In *APBC '04: Proceedings of the second conference on Asia-Pacific bioinformatics*, pages 297–302, Darlinghurst, Australia, Australia, 2004. Australian Computer Society, Inc.

[34] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[35] G. Karypis, E.-H. Han, and V. Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *IEEE Computer*, 32(8):68–75, August 1999.

[36] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[37] George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar. Multilevel hy-

pergraph partitioning: Applications in VLSI domain. In *Proceedings of the Design and Automation Conference*, pages 526–529, 1997.

[38] J. Kittler and F. Roli, editors. *Multiple Classifier Systems*. LNCS Vol. 2634, Springer, 2002.

[39] A. Kraskov, H. Stögbauer, R. G. Andrzejak, and P. Grassberger. Hierarchical clustering using mutual information. *EPL (Europhysics Letters)*, 70(2):278, 2005.

[40] L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, Hoboken, NJ, 2004.

[41] L. I. Kuncheva and S. T. Hadjitodorov. Using diversity in cluster ensemble. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 1214–1219, 2004.

[42] H. O. Lancaster. The chi-squared distribution. 1969.

[43] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *In Neural Information Processing Systems*, pages 556–562. MIT Press, 2000.

[44] T. Li and C. Ding. Weighted consensus clustering. In *Proceedings of Eighth SIAM International Conference on Data Mining*, pages 798–809, 2008.

[45] T. Li, C. Ding, and M. Jordan. Solving consensus and semi-supervised clustering problems using non-negative matrix factorization. In *Proceedings of Eighth IEEE International Conference on Data Mining*, pages 577–582, 2007.

[46] M. Meila. Comparing clusterings by the variation of information. In *Proceedings of Conference on Learning Theory*, pages 173–187, 2003.

[47] Marina Meilă. Comparing clusterings—an information based distance. *J. Multivar. Anal.*, 98(5):873–895, May 2007.

[48] Marina Meilă. Local equivalences of distances between clusterings–a geometric perspective. *Mach. Learn.*, 86(3):369–389, March 2012.

[49] S. Merugu and J. Ghosh. A distributed learning framework for heterogeneous data sources. In *Proc. Knowledge Discovery and Data Mining*, pages 208–217, 2005.

[50] B. Mirkin. *Mathematical Classification and Clustering*. Kluwer, 1996.

[51] S. Monti, P. Tamayo, J. Mesirov, and T. Golub. Consensus clustering-a resampling-based method for class discovery and visualization of gene expression microarray data. In *Journal of Machine Learning*, pages 52: 91–118, 2003.

[52] N. Nguyen and R. Caruana. Consensus clusterings. In *Proceedings of International*

*Conference on Data Mining*, pages 607–612, 2007.

[53] Xuan Vinh Nguyen, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11:2837–2854, 2010.

[54] N. C. Oza and K. Tumer. Classifier ensembles: Select real-world applications. *Inf. Fusion*, 9:4–20, January 2008.

[55] K. Punera and J. Ghosh. Consensus based ensembles of soft clusterings. In *Proc. MLMTA'07 - Int'l Conf. on Machine Learning: Models, Technologies & Applications*, 2007.

[56] Xavier Sevillano, Germán Cobo, Francesc Alías, and Joan Claudi Socoró. Feature diversity in cluster ensembles for robust document clustering. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 697–698, New York, NY, USA, 2006. ACM.

[57] A. Sharkey. *Combining Artificial Neural Nets*. Springer-Verlag, 1999.

[58] Vikas Singh, Lopamudra Mukherjee, Jiming Peng, and Jinhui Xu. Ensemble clustering using semidefinite programming with applications. *Mach. Learn.*, 79(1-2):177–200, May 2010.

[59] A. Strehl and J. Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3 (Dec):583–617, 2002.

[60] Alexander Strehl and Joydeep Ghosh. A scalable approach to balanced, high-dimensional clustering of market-baskets. In *Proc. HiPC 2000, Bangalore*, volume 1970 of *LNCS*, pages 525–536. Springer, December 2000.

[61] S. Swift, A. Tucker, V. Vinciotti, and N. Martin. Consensus clustering and functional interpretation of gene-expression data. In *Genome Biology 5:R94*, 2004.

[62] Stephen Swift, Allan Tucker, Veronica Vinciotti, Nigel Martin, Christine Orengo, Xiaohui Liu, and Paul Kellam. Consensus clustering and functional interpretation of gene-expression data. In *Genome Biology;5(11):R94*, 2004.

[63] A. Topchy, A. Jain, and W. Punch. A mixture model for clustering ensembles. In *Proceedings of SIAM International Conference on Data Mining*, pages 379–390, 2004.

[64] K. Tumer and J. Ghosh. Analysis of decision boundaries in linearly combined neural classifiers. *Pattern Recognition*, 29(2):341–348, 1996.

[65] K. Tumer and J. Ghosh. Robust order statistics based ensembles for distributed data mining. In Hillol Kargupta and Philip Chan, editors, *Advances in Distributed and Parallel Knowledge Discovery*, pages 85–110. AAAI Press, 2000.

[66] Fei Wang, Xin Wang, and Tao Li. Generalized cluster aggregation. In *Proc. of IJCAI'09*, pages 1279–1284, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[67] H. Wang, H. Shan, and A. Banerjee. Bayesian cluster ensembles. In *Proceedings of the Ninth SIAM International Conference on Data Mining*, pages 211–222, 2009.

[68] Pu Wang, Carlotta Domeniconi, and Kathryn Laskey. Nonparametric bayesian clustering ensembles. In *Machine Learning and Knowledge Discovery in Databases*, volume 6323 of *Lecture Notes in Computer Science*, chapter 28, pages 435–450. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2010.

[69] Matthijs Warrens. On similarity coefficients for 2x2 tables and correction for chance. *Psychometrika*, 73(3):487–502, September 2008.

[70] Pakaket Wattuya, Kai Rothaus, J.-S. Prassni, and Xiaoyi Jiang. A random walker based approach to combining multiple segmentations. In *ICPR'08*, pages 1–4, 2008.

[71] Junjie Wu, Jian Chen, Hui Xiong, and Ming Xie. External validation measures for k-means clustering: A data distribution perspective. *Expert Syst. Appl.*, 36(3):6050–6061, 2009.

[72] Y. Yang and M.S. Kamel. An aggregated clustering approach using multi-ant colonies algorithms. 39:109–117, July 2006.

[73] Yanfang Ye, Tao Li, Yong Chen, and Qingshan Jiang. Automatic malware categorization using cluster ensemble. In *Knowledge Discovery and Data Mining '10: Proceedings of the 16th ACM SIGKnowledge Discovery and Data Mining international conference on Knowledge discovery and data mining*, pages 95–104, New York, NY, USA, 2010. ACM.

[74] H.S. Yoon, S.Y. Ahn, S.H. Lee, S.B. Cho, and J.H. Kim. Heterogeneous clustering ensemble method for combining different cluster results. In *Proceedings of BioDM 2006, Lecture Notes in Computer Science*, volume 3916, pages 82–92, 2006.

[75] S. Zhong and J. Ghosh. A unified framework for model-based clustering. *Jl. Machine Learning Research (Journal of Machine Learning Research)*, 4:1001–1037, 2003.