Voting-Based Consensus of Data Partitions

by

Hanan G. Ayad

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2008

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Hanan G. Ayad

Abstract

Over the past few years, there has been a renewed interest in the consensus problem for ensembles of partitions. Recent work is primarily motivated by the developments in the area of combining multiple supervised learners. Unlike the consensus of supervised classifications, the consensus of data partitions is a challenging problem due to the lack of globally defined cluster labels and to the inherent difficulty of data clustering as an unsupervised learning problem. Moreover, the true number of clusters may be unknown. A fundamental goal of consensus methods for partitions is to obtain an optimal summary of an ensemble and to discover a cluster structure with accuracy and robustness exceeding those of the individual ensemble partitions.

The quality of the consensus partitions highly depends on the ensemble generation mechanism and on the suitability of the consensus method for combining the generated ensemble. Typically, consensus methods derive an ensemble representation that is used as the basis for extracting the consensus partition. Most ensemble representations circumvent the labeling problem. On the other hand, voting-based methods establish direct parallels with consensus methods for supervised classifications, by seeking an optimal relabeling of the ensemble partitions and deriving an ensemble representation consisting of a central aggregated partition. An important element of the voting-based aggregation problem is the pairwise relabeling of an ensemble partition with respect to a representative partition of the ensemble, which is refered to here as the voting problem. The voting problem is commonly formulated as a weighted bipartite matching problem.

In this dissertation, a general theoretical framework for the voting problem as a multi-response regression problem is proposed. The problem is formulated as seeking to estimate the uncertainties associated with the assignments of the objects to the representative clusters, given their assignments to the clusters of an ensemble partition. A new voting scheme, referred to as cumulative voting, is derived as a special instance of the proposed regression formulation corresponding to fitting a linear model by least squares estimation. The proposed formulation reveals the close relationships between the underlying loss functions of the cumulative voting and bipartite match-

ing schemes. A useful feature of the proposed framework is that it can be applied to model substantial variability between partitions, such as a variable number of clusters.

A general aggregation algorithm with variants corresponding to cumulative voting and bipartite matching is applied and a simulation-based analysis is presented to compare the suitability of each scheme to different ensemble generation mechanisms. The bipartite matching is found to be more suitable than cumulative voting for a particular generation model, whereby each ensemble partition is generated as a noisy permutation of an underlying labeling, according to a probability of error. For ensembles with a variable number of clusters, it is proposed that the aggregated partition be viewed as an estimated distributional representation of the ensemble, on the basis of which, a criterion may be defined to seek an optimally compressed consensus partition.

The properties and features of the proposed cumulative voting scheme are studied. In particular, the relationship between cumulative voting and the well-known co-association matrix is highlighted. Furthermore, an adaptive aggregation algorithm that is suited for the cumulative voting scheme is proposed. The algorithm aims at selecting the initial reference partition and the aggregation sequence of the ensemble partitions the loss of mutual information associated with the aggregated partition is minimized. In order to subsequently extract the final consensus partition, an efficient agglomerative algorithm is developed. The algorithm merges the aggregated clusters such that the maximum amount of information is preserved. Furthermore, it allows the optimal number of consensus clusters to be estimated.

An empirical study using several artificial and real-world datasets demonstrates that the proposed cumulative voting scheme leads to discovering substantially more accurate consensus partitions compared to bipartite matching, in the case of ensembles with a relatively large or a variable number of clusters. Compared to other recent consensus methods, the proposed method is found to be comparable with or better than the best performing methods. Moreover, accurate estimates of the true number of clusters are often achieved using cumulative voting, whereas consistently poor estimates are achieved based on bipartite matching. The empirical evidence demonstrates that the bipartite matching scheme is not suitable for these types of ensembles.

Acknowledgments

I wish to start by expressing my deep gratitude to God for the wonder that can endlessly inspire interest and a quest for knowledge.

My aspiration to pursue this research could not have become a reality without the valuable contributions of a number of supportive people, for whom I will always be grateful.

I am grateful to my supervisor, Prof. Mohamed Kamel, for introducing me to an interesting research area that crosses the boundaries of several disciplines and for patiently fostering my development as a researcher. I sincerely thank him for his committment to giving me advice, support, and encouragement; for his valuable insights and feedback; and for his constant dedication to discussing this research.

I am grateful to Prof. Hugh Chipman for his insightful teaching of Statistical Learning, and for kindly accepting to serve on my committee, despite the difficult logistics due his move to Acadia University, early on in my program. I wish to thank him for the time he took to meet with me during his visits to Waterloo, for the numerous comments and suggestions he gave me, and for a useful discussion on the multi-response regression formulation proposed in this thesis.

I am grateful to the ECE members of my committee: Prof. Otman Basir and Prof. William Bishop, and to my external examiner Prof. Ludmila Kuncheva for their insightful feedback and suggestions on the thesis. I also wish to sincerely thank Prof. Bishop for offering many useful comments on the writing of the thesis.

I am thankful for the financial support of the Natural Sciences and Engineering Research Council (NSERC) of Canada, the Ontario Graduate Scholarship (OGS) program, the Faculty of Engineering and the Graduate Studies Office at the University of Waterloo, and the Learning Objects Repository Network (LORNET).

I am grateful to my husband Ossama El Badawy, an alumni of the Pattern Analysis and Machine Intelligence research group, for the numerous discussions I had with him and the technical help he gave me, and for his amazing support and encouragment of my research work.

I am grateful to Douglas Harder for the knowledge and skills I gained by working with him as a teaching assistant during several semesters of my Ph.D. program. His knowledge and tireless dedication will always be very inspiring.

I am also grateful to Prof. Paul Fieguth for his insightful teaching of Pattern Recognition.

I wish to thank the member and alumni of the PAMI group. I am especially thankful for the useful discussions and memorable interactions during my PhD years with Moataz El Ayadi, Masoud Makrehchi, Khaled Hammouda, Shady Shehata, Abbas Ahmadi, Yanmin Sun, Rozita Dara, Kong Wai-Kin Adams, Yu Sun, Ali Ahmed, and Ibrahim El-Rube.

I am thankful to the administrative staff for their great help during my program. In particular, I wish to thank Wendy Boles, Lisa Hendel, Heidi Campbell, Sue Havitz, and Karen Critchley.

I wish to thank my father-in-law for his support and his motivating interest in science.

I am grateful to my mother and father; their great support and their pride and joy in my humble accomplishments is very generous.

I am indebted to my daughter Sarah for enduring my long working days and nights.

Dedication

To Ossama and Sarah.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This introductory chapter begins with an overview of the data clustering problem, in Sec. 1.1, where the issues leading to the idea of seeking a consensus clustering are highlighted. In Sec. 1.2, the problem of reconciling an ensemble of partitions is introduced, and a distinct class of consensus methods referred to as voting-based methods is emphasized. In Sec. 1.4, some general notations are presented. The contributions of the thesis are outlined in Sec. 1.3, and in Sec. 1.5, the thesis organization is summarized.

1.1 Data Clustering

The goal of data clustering is the discovery of a meaningful and consistent cluster structure for a given collection of *data objects*, where the objects are typically described by a number of *input descriptor variables*. The data objects are also known as *data samples* or *observations*, and the set of input variables is also known as the *feature space*. Data clustering addresses a fundamental problem in exploratory data analysis, where the basic idea is to identify natural groups in a dataset by assigning the objects that are inherently similar to the same cluster and those that are inherently dissimilar to different clusters. The problem has a been extensively studied for

several decades in the areas of pattern recognition, machine learning, applied statistics, as well as in communications and information theory [1–9]. The problem arises in numerous fields of applications including data mining, document clustering, bio-informatics, image analysis and segmentation, data compression, and data classification.

Data clustering is also known as *unsupervised learning*. The term "unsupervised" is given in contrast with the related *supervised learning* problem, whereby a training (learning) dataset, which includes a target output variable for the objects, is available and used to train/learn a predictive model. The predictive model can then be used to predict the value of the output variable for new samples with unknown or missing values of the output variable. Essentially, supervised learning seeks to specify a mathematical model that best represents the relationship between the input (descriptor) variables and an output response variable, possibly based on a presumed functional form such as linear, quadratic, specific probability functions, or other non-parametric models. It is important to note that supervised learning is a general term that encompasses not only the problem of learning a categorical (class) variable, but also the problem of learning a numerical output variable or multiple output variables. In particular, learning a numerical output variable corresponds to the classical *regression* problem [10]. Learning a categorical variable is known as data classification.

In a data clustering problem, all the available data objects are unlabeled, and one is seeking the extraction of an output categorical variable whose values depend on the input variables. Finding the model that best fits the data can shed light on the natural relationships between the objects and on the underlying population model. A clustering solution for a set of n data objects corresponds to a *partition* of the n objects into k clusters, such that each object is assigned a symbolic cluster label, where usually $k \ll n$. A clustering may also correspond to a hierarchical sequence of partitions, referred to as a *dendrogram*, where k takes sequential values in $\{n, \ldots, 1\}$, corresponding to each level of the hierarchy. The true or meaningful number of clusters in the data may be unknown, in which case, the problem of determining an optimal number of clusters for the data needs to be addressed.

Generally, data clustering represents a challenging combinatorial optimization problem. The number of ways of partitioning a set of n objects into k non-empty clusters is a Stirling set number of the second kind, which is of the order $k^n/k!$ [9]. Thus, it is computationally intractable for large problem sizes to exhaustively examine all possible clustering solutions. For solving a clustering problem, a criterion is defined, and approximation or local search algorithms are devised to find a clustering solution that best fits the data, based on the presumed clustering model. Multiple different solutions are possible. Moreover, the inherent difficulty of the clustering problem is not only due to the computational aspects and to the fact that different local optima are possible. It is also the unsupervised nature of data clustering and the possible lack of knowledge of the true number of clusters, that makes it particularly difficult. Clustering algorithms often lead to a successful discovery of optimal cluster structures, if the data fit well to the presumed model. However, if the data is not correctly modeled by the specified criterion, the discovered cluster structures are generally inadequate.

As increasingly complex and large data are common in current applications, it is more problematic to select a clustering criterion that leads to extracting meaningful cluster structures. Clusters may have complex shapes, highly unbalanced sizes, different densities, and possible overlaps. Furthermore, most current data collections have high dimensional feature spaces, where in some cases, the number of features can be thousands or tens of thousands, as in the case text and micro-array data. Such very high dimensionality leads to the problem known as the curse of dimensionality, where cluster structures are hidden in the huge feature space.

To overcome the inherent difficulties of data clustering and deal with a variety of challenging data, complex clustering criteria in conjunction with computationally fast algorithms are required. In particular, the design of consensus methods that produce a consensus clustering for a generated ensemble of partitions represents a promising direction. It provides an advanced step for the underlying learning task where an ensemble instead of an individual learner is employed in search for a consistent cluster structure reflected by a consensus partition. Furthermore, complex cluster structures may be viewed as being composed of combinations of simpler or partial structures.

1.2 Consensus of Partitions

The multiplicity of clustering solutions for a given dataset is an issue that has long been known in the data clustering literature. The consensus of multiple data partitions was investigated within a body of research that addressed the problem of *comparison and consensus of data classifications* [11–15]. An overview of related work in this early literature is presented in Ch. 2. In this work, the term "classifications" refers to a wide variety of categorization structures including partitions, dendrograms, n-trees, ordered trees, phylogenetic trees, unrooted trees, and graphs, and the consensus problem was addressed for various types of structures. Applications included taxonomic and systematic research, bio-mathematics, and quantitative social sciences [16]. Traditionally, the consensus problem for partitions has been studied in the fields of discrete mathematics and theoretical computer science, classification and numerical taxonomy, and applied mathematics in humanities and quantitative sociology. Classical approaches to the consensus problem for partitions are categorized as axiomatic, constructive, and combinatorial optimization methods [15].

During the past few years, there has been a renewed interest in the problem of combining an ensemble of partitions, also known as a cluster ensemble. Emerging interest in the problem of generating and combining cluster ensembles is primarily motivated by the advances in the related area of combining multiple supervised classifications. This area has been mainly developed in the fields of machine learning, applied statistics, and pattern recognition. An array of classifier ensemble methods was developed including bagging [17], boosting [18], randomized decision forests [19, 20], additive logistic regression [21], and the random subspace method [20].

Cluster ensembles have been investigated for achieving various objectives such as improving clustering accuracy over a single data clustering, allowing the discovery of arbitrarily-shaped cluster structures [22–27], reducing the instability of a clustering algorithm due to noise, outliers [28], or to randomized algorithms [22], reusing pre-existing clusterings (knowledge reuse) [29], exploring random feature subspaces [29, 30] or random projections [30] for high dimensional data,

exploiting weak clusterings such as splitting the data with random hyperplanes [30], estimation of confidence in cluster assignments for individual observations [31], and clustering in distributed environments including feature or object distributed clustering [29]. Combination of cluster ensembles have been investigated for several application domains such as cluster analysis for gene expression data [31], distributed data mining [29, 32], and image segmentation [28].

A fundamental goal of consensus methods for cluster ensembles is to find a consensus partition that optimally summarizes an ensemble and reveals a cluster structure with accuracy and robustness exceeding the individual ensemble partitions. In the case of supervised learning, the design of combination rules is relatively simple. For instance, aggregation can be directly applied [17]. In regression, the aggregation can be obtained by simple averaging of the individual learners. For classifiers, the aggregation may be obtained via plurality voting; the assignments of the data objects to classes by individual classifiers are viewed as votes, and aggregated vote ratios may be used to assign objects to their highest voted class. However, the aggregation of multiple partitions is a challenging problem [33, 34]. Unlike classifiers, different clusters are distinguished by arbitrary symbols rather than globally defined class labels, usually numbered as $\{1, \ldots, k\}$. That is, all $k!$ permutations of the cluster labels for a given partition represent the same partition. Furthermore, the ensemble partitions can have a variable number of clusters. This lack of pre-defined classes in the data, or the number thereof, causes a relabeling problem, in addition to the problem of finding an optimal number of clusters.

It is noted that a consensus method typically derives an internal data representation from the ensemble of partitions, which is referred to here as an *ensemble representation*. It is on the basis of the ensemble representation that the optimal consensus partition is determined. The following ensemble representations appear in recent work. Proximity-based representations were considered including the co-association matrix [25, 29, 31, 35–37] and the graph-based representations proposed in [29]. A categorical feature-space representation, where each variable corresponds to an ensemble partition was considered in [38]. In [39], we considered a distributional data representation, reflecting the uncertainties associated with the assignment of the data objects to a set

of reference clusters, as input to a subsequent information-theoretic clustering algorithm.

In fact, the consensus problem can itself be viewed as a clustering problem, whereby the data partitioning is based on the representation derived from the ensemble. Recent consensus methods may be roughly classified into proximity-based methods such as [25, 29, 31, 40], methods based on a (categorical) feature-space representation of the objects [38, 41], and methods based on voting (ensemble relabeling) as in [28, 31, 33, 39, 42, 43]. This categorization is elaborated on in Ch. 2.

The first two types of consensus methods derive an ensemble representation that side-steps the relabeling problem. On the other hand, voting-based consensus methods, seek to establish a parallel approach to the aggregation of supervised learners [17, 18, 21], by searching for an optimal relabeling of the ensemble partitions. The ensemble relabeling enables the computation of an aggregated partition. That is, unlike other consensus methods, voting-based methods derive an ensemble representation consisting of a central aggregated partition.

As pointed out in [33,34], the problem of relabeling and aggregating an ensemble of partitions is computationally challenging as it requires the simultaneous optimization of relabeling the ensemble partitions with respect to the representative (aggregated) partition of the ensemble and of the aggregated partition with respect to the ensemble partitions. Several efficient algorithms are proposed in recent literature [28,31,33,39,42,43], as detailed in Ch. 3. An important element of the ensemble relabeling problem is the pairwise (partial) relabeling of the ensemble partitions, referred to here as the *voting problem*. The problem addresses the optimal relabeling of each ensemble partition with respect to a representative partition of the ensemble. It is commonly formulated as a weighted bipartite matching problem [28, 31, 33, 42, 43].

Note that the term "voting" essentially refers to the assignment of representative cluster labels to the data objects, as derived from an ensemble partition. The common bipartite matching scheme corresponds to permuting the cluster labels of an ensemble partition to optimally match the labels of the representative partition. That is, it corresponds to a binary one-to-one relabeling and hence it is also referred to here as a binary voting.

1.3 Contributions

In what follows is an overview of the contributions of this dissertation.

1. A general theoretical framework for the voting problem as a multi-response regression problem is proposed, whereby the relabeling of an ensemble partition is formulated as the problem of estimating the uncertainties associated with the assignments of the object to representative clusters, given their assignments to the clusters of an ensemble partition.

    - A new voting scheme, referred to as cumulative voting, is derived as a special instance of the proposed regression formulation, which corresponds to fitting a linear model by least squares estimation.
    - An iterative aggregation algorithm referred to as `Vote` is applied as a general algorithm, with variants referred to as `cVote` and `bVote`, corresponding to cumulative voting and bipartite matching, respectively. A simulation-based analysis is conducted to compare `cVote` and `bVote`, demonstrating that `bVote` is more suitable than `cVote` for a particular partition generation model considered in [43], whereby the ensemble partitions are generally uniform, with equal number of clusters, and are generated as noisy permutations of an underlying labeling, according to a probability of error.
    - For other types of ensembles, such as ensembles with a variable number of clusters, it is proposed that the consensus partition be considered at multiple granularities (i.e., number of clusters) and a principled information-theoretic approach be applied for extracting an optimally summarized consensus partition.

2. A study of the properties of the proposed cumulative voting scheme is presented.

    - A relationship is derived between cumulative voting and the co-association matrix, which is a fundamental aggregated representation for partition ensembles. To describe this relation, an un-normalized variant of the cumulative voting scheme is defined.

- An entropy preserving property for the cumulative voting scheme is noted. This property is utilized to introduce an adaptive aggregation algorithm referred to as `Ada-cVote`. Unlike the `Vote` algorithm, which randomly selects the initial reference partition and considers the ensemble partitions in a random order, `Ada-cVote` selects the initial reference partition and the aggregation sequence of the ensemble partitions such that the loss in the mutual information associated with the estimated aggregated distribution is minimized. This leads to a more "informative" aggregated partition.

3. For ensembles with a relatively large and variable number of clusters, an interpretation of the aggregated partition as a distributional representation of the ensemble is proposed. The aggregated partition is considered the most granular consensus partition available (consisting of $\bar{k}$ clusters), and consensus partitions at coarser levels are sought.

   - An efficient agglomerative algorithm is proposed for extracting a hierarchy of consensus partitions, each representing an optimally compressed k-cluster summary of the aggregated representation that preserves the maximum amount of relevant information, where $k \in \{\bar{k}, \ldots, 1\}$. Furthermore, an approach to estimating an optimal number of consensus clusters is applied.
   - The proposed algorithm, which applies the Jensen-Shannon divergence and the average link hierarchical clustering (referred to as `JS-Alink`), is applied in conjunction with `Ada-cVote` to give an overall two-stage consensus algorithm referred to as ACV. When ACV is used with a pre-determined number of consensus clusters k, it is referred to as ACV-k. To compare the cumulative voting scheme with bipartite matching, the `JS-ALink` algorithm is also applied in conjunction with `bVote` to give another variant of a two-stage consensus algorithm referred to as BV. Also, when BV is given a desired number of consensus clusters, it is referred to as BV-k.
   - An important advantage of the proposed voting-based consensus method is that it is

computationally more efficiency than co-association based methods. It is characterized by a linear complexity in the number of data objects $O(n)$, whereas co-association based methods are $O(n^2)$.

4. An empirical study is conducted using several artificial and real-world datasets with various characteristics and difficulties, including a text data characterized by a very high dimensional feature space. Furthermore, several recent consensus algorithms are applied for a comparative evaluation. An ensemble generation technique that produces partitions with a variable and relatively large number of clusters is adopted. Empirical evidence is presented demonstrating that the ACV-k algorithm can find substantially more accurate consensus partitions compared to BV-k, and is either comparable or better than the winner(s) among all other consensus algorithms. Furthermore, accurate estimates of the optimal number of clusters are often achieved using ACV, whereas consistently poor estimates are achieved using BV. The empirical results demonstrates that the bipartite matching is not suitable for this type of ensemble, whereas cumulative voting can be successful.

1.4 Notations

The following general convention is used. Scalars are written in lowercase letters. Vectors are denoted by boldface lowercase letters, and matrices are denoted by boldface uppercase case letters. The transpose of a matrix $\mathbf{U}$ is denoted by $\mathbf{U}^T$. Random variables are denoted by uppercase letters. Sets are denoted by capital calligraphic letters, whereas members of a set are denoted by lowercase letters. A set of data objects is written as $\mathcal{X} = \{x_1, \ldots, x_n\}$, where x_j denotes the label of a member of $\mathcal{X}$. It is noted that, at the ensemble generation phase, each data object is a point in $\Re^p$ represented by the vector $\mathbf{x}_j$. However, in the consensus phase, the objects are not dealt with in their feature space representation. Instead, they are viewed as members of the set $\mathcal{X}$. The set of cluster labels is denoted as $\mathcal{C} = \{c_1, \ldots, c_k\}$. The discrete random variables taking

values in $\mathcal{X}$ and $\mathcal{C}$ are denoted by X and C. The conditional distribution $p(c|x)$ is defined for the discrete variables C and X and is estimated through the aggregation of the ensemble.

1.5 Thesis Organization

The thesis is organized as follows. In Ch. 2, a review of related work on consensus methods for partitions is presented. In Ch. 3, the proposed framework and the cumulative voting scheme are described, the voting-based aggregation problem for partitions is addressed, and a preliminary simulation-based analysis is presented to compare the aggregation based cumulative voting versus bipartite matching. In Ch. 4, the properties and features of the proposed cumulative voting scheme are studied, and an adaptive aggregation algorithm is introduced. In Ch. 5, the problem of extracting an optimally compressed summary of the aggregated representation of the ensemble is addressed and an empirical study is presented to validate the proposed consensus algorithms based on cumulative voting. Finally, the conclusions and future directions are outlined in Ch. 6.

Chapter 2

Review of Related Work

In this chapter, a review of related work on consensus methods for partitions is presented. The idea of consensus is a general multi-disciplinary topic that been extended to the area of reconciling partitions early on. A brief overview of early literature is outlined in Sec. 2.1. However, an in-depth review of the multi-disciplinary work on consensus theory is outside the scope this thesis. The interested reader is referred to a recent book on the theory of consensus in group choice and bio-mathematics by Day and McMorris [16].

During the past few years, interest re-emerged in the consensus clustering problem for a partition ensemble. Recent work has been primarily motivated by the preceding advances in the area of combining multiple supervised classifications. A detailed review of the relatively mature area of combining multiple classifiers is beyond the scope of this thesis. The interested reader is referred to a recent book by Kuncheva [44]. In Sec. 2.2, a taxonomy and a survey of recently developed consensus methods for partitions is presented. Furthermore, the different ensemble generation mechanisms that have been developed in recent work are discussed in Sec. 2.3, and an overview of some analytical studies for cluster ensembles is presented in Sec. 2.4.

2.1 Early Literature

This section presents a brief chronology of the consensus problem and an outline of early work on consensus of partitions.

2.1.1 Introduction

Originally, the consensus problem arose in the social sciences where it addresses the aggregation of a set of *individual preferences* into a single *social preference*. In this context, consensus methods can be traced back to the late eighteenth century to the works of Borda and Condorcet who formulated voting methods and developed voting systems known as the Borda count method, in 1770, and the Condorcet method, in 1785, respectively.

In the second half of the twentieth century, the advances in what is referred to as the *theory of consensus*, and its extension to other scientific disciplines, were motivated by Arrow's theorem introduced in his 1951 Ph.D. thesis (which later contributed to his 1972 Nobel Prize in Economics) [16]. Arrow introduced an axiomatic paradigm in group choice, where the goal is to aggregate a set of individual rankings into a single group ranking. He established the impossibility of existence of a consensus rule that satisfies a set of reasonable fairness axioms. This contradictory result was used to achieve the plurality rule and other non-dictatorial rules through appropriate weakening of the axioms [16].

While Arrow's work was in the area of social choice theory, dealing with the aggregation of models of preferences, the aggregation of other models, including classification models such as partitions, phylogenetic trees, and different tree and graph structures, was a general problem arising in other areas of science and technology, such as taxonomic and systematic research and bio-mathematics [16]. Hence, Arrow's results were extended to the aggregation of classification models. In 1975, Mirkin [11] introduced an impossibility theorem for reconciling partitions of a set. Interest in the problem of consensus classifications grew steadily. In 1986, a special issue of the journal of classification investigated methodologies for the comparison and consensus of

classifications [14].

The problem of reconciling partitions was defined as the problem of finding a consensus partition that summarizes a *profile* or a *family* of partitions in some meaningful sense [15]. Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ denotes a set of data objects and let a profile of partitions be denoted as $\mathcal{P} = \{\pi_1, \ldots, \pi_b\}$, where π_i is a member of $\mathcal{P}$, and b is the cardinality of the $\mathcal{P}$, $b = |\mathcal{P}|$. Barthélemy and Leclerc [15] point out that a consensus problem arises when we have one of the following cases. (1) There are b partitioning methods providing approximations of the desired classification, (2) a profile consists of b categorical variables $C^1, \ldots, C^b$ describing the set of data objects and leading to a search for a partition that is closest to the profile, in the sense of the statistical idea of a central value, or (3) there is a profile representing partitions from measurements at times $t, t+1, \ldots, t+b-1$, where in such a case, the notion of a *moving consensus* corresponds to a smoothing of the series of partitions (or categorical variables).

2.1.2 Classical Approaches to Reconciling Partitions

In addressing the problem of reconciling partitions, three overlapping approaches were developed early on [15], as described below. The third approach, being the most related to current research on cluster ensembles, is elaborated upon.

1. The first, designated as *axiomatic* [11, 45–47], is related to Arrow's approach and is concerned with deriving possibility/impossibility theorems on the existence and uniqueness of consensus partitions satisfying specific conditions.

2. The second, designated as *constructive*, specifies rules for constructing a consensus, such as the Pareto rule, also known as the *strict consensus rule*, whereby two objects occur together in a consensus if and only if they occur together in all the individual partitions.

3. The third approach, designated as *combinatorial optimization*, considers a criterion measuring the *remoteness* $R(\pi, \mathcal{P})$ of any partition π of $\mathcal{X}$ to the given profile $\mathcal{P}$, and defines

the optimal consensus partition as the partition solution of the problem in Eq. 2.1, where Π represents the set of all possible partitions of $\mathcal{X}$.

$$\min_{\pi \in \Pi} R(\pi, \mathcal{P}) \tag{2.1}$$

The approach is related to the notion of a central value in statistics, and goes back to Régnier [48], who used the term *partitions centrales* and considered the sum of the squared distances, as an optimization criterion. Régnier formulates the problem of finding a central partition [48], defined as a partition with maximum similarity to the input partitions. For each partition π^i of the objects, an $n \times n$ boolean matrix $\mathbf{M}^i = [m^i_{jg}]$ is defined, where $m^i_{jg} = 1$ if objects j and g belong to the same cluster of partition π^i. The distance between two partitions π^i and π^r is defined as $d(\mathbf{M}^i, \mathbf{M}^r) = \sum_{j=1}^{n} \sum_{g=1}^{n} (m^i_{jg} - m^r_{jg})^2$. For a profile of b partitions, an aggregate matrix is defined as $\mathbf{M} = \sum_{i=1}^{b} \mathbf{M}^i$. The central partition is defined as the partition π^* with corresponding boolean matrix $\mathbf{M}^*$ that minimizes the criterion $\sum_{i=1}^{b} d(\mathbf{M}^*, \mathbf{M}^i)$. Régnier [48] also presented a theoretical study of the convergence of central partitions based on a metric and a probability measure on the space of partitions.

Barthelemy and Leclerc [15] note that, in general, any distance function d on Π may be considered, where the remoteness of π to $\mathcal{P}$, as given in Eq. 2.2, is defined as the sum of the distances in the median case, or the sum of the squares of the distances, in the center case as in [48]. Various metrics on partitions were studied in [49, 50].

$$\min_{\pi \in \Pi} \sum_{i=1}^{b} d(\pi, \pi^i) \tag{2.2}$$

The method is referred to as the *median procedure for partitions* [15,51]. For a given distance function d, the procedure is referred to as the d-median procedure, and the solution of the problem in Eq. 2.2 is referred to as the d-median partition. The problem is viewed as a clique partitioning problem and the algorithmic approaches are classified into four classes: exact

methods (such as branch and bound, which is suitable for only very small n), relaxation methods, hill climbing heuristics, and meta-heuristics [15].

2.2 Review of Recent Consensus Methods

In this section, recent consensus clustering methods are organized into distinct classes, and a survey is presented. It is noted that a primary design element of a consensus clustering algorithm is the aggregate representation that is constructed for the ensemble of partitions. In other words, a consensus method generally derives an aggregate representation of the ensemble, which is used as the basis for extracting a consensus clustering. Hence, a general taxonomy of consensus methods can be developed according to the different types of their aggregate ensemble representations. An overview and a schematic diagram of the proposed taxonomy is presented in Sec. 2.2.1, and detailed descriptions are given in Secs. 2.2.2, 2.2.3, and 2.2.4.

2.2.1 Consensus Methods: A Taxonomy

Three main types of aggregated representations can be identified in recent consensus methods, where in turn, each representation can be further sub-divided into other subtypes or according to the specific ways in which it is used to extract a consensus clustering. Figure. 2.1 depicts the proposed taxonomy, with many recent consensus algorithms being identified, including the algorithms proposed in this thesis. As shown in Fig. 2.1, one major class consists of consensus methods that induce a similarity-based structure from a given ensemble. A second class include methods that view the ensemble partitions in a categorical feature-space representation. Finally, a third class consists of methods that derive a voting-based aggregated partition, which is a *soft* partition, to represent a given ensemble.

A similarity-based structure can be subdivided into different subtypes, as follows. One basic structure represents pairwise similarities between each pair of data objects. This structure corresponds to an undirected graph where nodes represent objects and edges are weighted according to

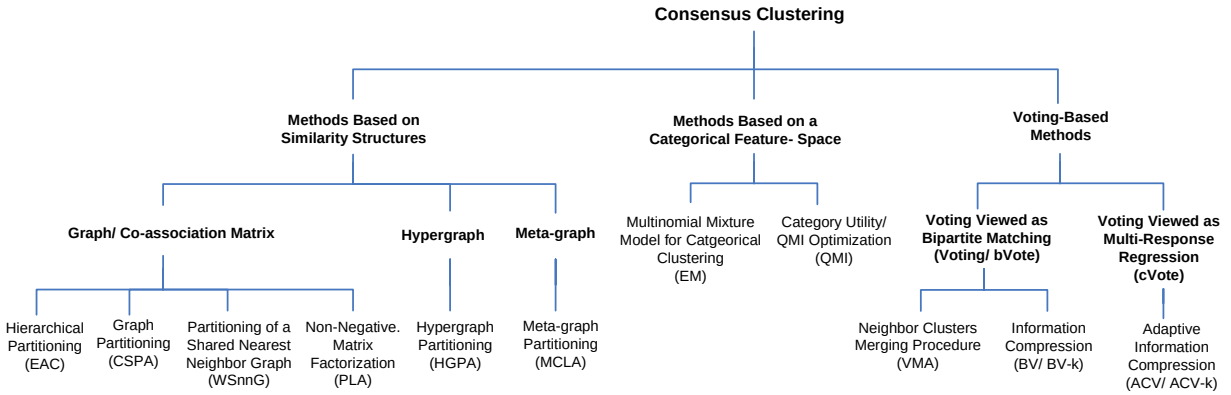

Figure 2.1: A taxonomy of consensus clustering methods based on the aggregate representation of the ensemble partitions. Several recent consensus algorithms are identified.

a defined measure of similarity. The graph can generally be represented by an $n \times n$ adjacency matrix. In the context of cluster ensembles, this adjacency matrix is also known as a co-association matrix, where similarities reflect the frequency of co-occurrence of each pair of objects in the same cluster throughout the ensemble. The co-association matrix has been utilized in a number of different ways to extract a consensus partition. For instance, a hierarchical linkage-based clustering is applied in the evidence accumulation clustering (EAC) algorithms [25]. Additionally, a graph partitioning algorithm referred to as cluster-based similarity partitioning algorithm (CSPA) is proposed in [29]. Furthermore, an approach based on constructing a weighted shared nearest neighbors graph (WSnnG) from the co-association values is developed in [26]. Moreover, a probabilistic label aggregation algorithm (PLA) is introduced in [37], which is based on non-negative matrix factorization [52] of the normalized co-association matrix.

A second similarity-based structure proposed in [29] is a hypergraph, where an edge represents multi-way similarity connecting multiple objects, instead of a pair. A consensus clustering is obtained by applying a hypergraph partitioning algorithm (HGPA) [29]. Additionally, a *meta*-graph similarity structure is proposed in [29] where edges represent similarity relations between pairs of clusters, rather than objects. In this case, (meta)-graph partitioning is applied to find a

consensus partition as given by the meta-clustering algorithm (MCLA) [29].

In a categorical feature representation of the objects, each feature corresponds to an ensemble clustering represented as a labeling vector. Methods based on this representation are as follows. A statistical model-based clustering using a mixture of multinomial distributions was proposed in [41], where a consensus partition is computed as a solution to the corresponding maximum likelihood problem using the EM algorithm. An alternative approach is proposed in [38], where the category utility function is defined as the criterion for the categorical clustering problem. The criterion is shown to be equivalent to the quadratic mutual information and the classical intra-class variance criteria. A consensus partition is extracted by transforming the categorical variables into standardized numerical variables and by applying the k-means algorithm as a mean-squared error algorithm [38]. The consensus algorithm is referred to as QMI.

It is noted that for methods based on similarity structures or on a categorical feature-space representation, the ensemble re-labeling problem is circumvented. Contrarily, when deriving a voting-based aggregated partition, the ensemble re-labeling problem is directly addressed. Voting-based consensus methods constitute the main focus of this dissertation. The ensemble re-labeling is a difficult problem. It is generally attacked via pairwise relabeling. The pairwise relabeling if also referred to here as the voting problem. It has been commonly formulated as a bipartite matching problem [28, 31, 33, 42, 43]. A voting-based aggregation algorithm where bipartite matching is applied (e.g. Voting [33]) is referred to here as bVote. When the ensemble partitions are optimally relabeled, an aggregated partition can be computed by averaging. A consensus partition can be obtained by assigning each object to its most voted cluster. If the desired number of consensus clusters is smaller than the number of aggregated clusters, a merging of cluster can be applied as described in the voting-merging algorithm (VMA) proposed in [53].

An alternative approach for addressing the voting problem, which is proposed in this thesis, uses a probabilistic scheme as introduced in [39], and is referred to as cumulative voting. A voting-based aggregation algorithm where cumulative voting is applied is referred to here as cVote. As explained in Chapter 3, the cumulative voting scheme can be viewed a special instance of a

more general framework whereby the voting problem is formulated as a multi-response regression problem. The consensus can be obtained by assigning each object to its most likely cluster. If the desired number of consensus clusters is smaller than the number of aggregated clusters, the aggregated partition can be viewed as a stochastic representation where information theoretic principles can be applied to extract a compressed consensus partition as described in [39], and later in this thesis. The information theoretic algorithms identified as BV, BV-k, ACV, and ACV-k in Fig. 2.1 will be described in detail in later chapters.

2.2.2 Similarity-Based Consensus Methods

Consensus methods that derive its primary aggregated representation as a similarity structure are described below in further details.

Graph/ Co-Association Matrix

The co-association matrix is a fundamental similarity-based representation that is similar to that used by Régnier [48], in the early literature on consensus partitions. One of its major advantages is that the labeling problem is circumvented, whereas a major disadvantage is the quadratic computational complexity in the number of objects $O(n^2)$ of co-association-based consensus methods, which makes it unattractive for very large datasets.

In [22–25], Fred and Jain propose an evidence accumulation clustering (EAC) as a co-association based consensus method. Each partition is viewed as an independent evidence on the pairwise objects co-associations, and the co-association matrix is viewed as the outcome of a voting (or evidence gathering) mechanism. The consensus partition is extracted by applying a hierarchical agglomerative clustering algorithm on the co-association matrix such as the single-link and average-link algorithms. To determine the natural number of clusters for a dataset, the notion of a cluster *lifetime* is defined as the range of threshold values on the dendrogram that correspond to a k-cluster partition. It is computed as the difference between the minimum threshold value

that corresponds to a k-partition and that which corresponds to a $k-1$-partition. The optimal number of clusters is the one with the longest cluster lifetime [25].

Dudoit and Fridlyand [31] develop bagging procedures for data clustering inspired by bagging in supervised learning [17]. One of their methods, referred to as "BaggClust2", is based on computing a dissimilarity matrix from the frequencies of co-occurrence (co-association) of each pair of objects in the same cluster. Subsequently, the dissimilarity matrix is used as input to a clustering method to extract the consensus partition. The Partitioning Around Medoids (PAM) clustering method of Kaufman and Rousseeuw [54] is used in [31].

Strehl and Ghosh [29] define the normalized mutual information (NMI) as a measure of agreement or similarity between two partitions. The optimal consensus partition is defined as the partition that maximizes the average normalized mutual information (ANMI) with the input partitions. To extract the consensus partition, different graph-based methods were introduced, where the graph partitioning algorithms of Karypis and Kumar [55, 56] were applied. The CSPA algorithm in [29] uses a graph structure that corresponds to a co-association matrix, where objects represent the graph nodes, and undirected edges are weighted by the co-association values. The graph partitioning algorithm METIS [55, 56] is applied to extract the consensus partition.

Monti, Tamayo, Mesirov, and Golub [40] construct a consensus matrix, similar to a co-association matrix. The matrix is used to aggregate the results of multiple random restarts of a clustering algorithm (such as k-means, model-based Bayesian clustering, or SOM) in conjunction with either bootstrap samples of the dataset or gene resampling (on gene expression data). The consensus matrix is also used as a visualization tool, by arranging it so that items belonging to the same cluster are adjacent to each other. Furthermore, a color gradient is associated to the 0-1 range of real numbers, so that white corresponds to 0, and dark red corresponds to 1. So, a matrix corresponding to a perfect consensus will be characterized by red blocks along the diagonal, on a white background. To find the number of clusters k that best fits the data, a consensus matrix is constructed for each $k = \{2, 3, \ldots, k_{\max}\}$, where the ensemble partitions are generated with k clusters each. The best number of clusters corresponds to the "cleanest" matrix (which

is a matrix containing mostly 0's or 1's). They propose a measure referred to as a consensus distribution which is computed based on the consensus matrix to help determine the number of clusters. When the best k is determined, the hierarchical average linkage clustering algorithm is applied on the corresponding consensus matrix and the k-cluster partition is extracted.

Ayad and Kamel [26,57] derive a graph representation referred to as Weighted Shared Nearest Neighbor Graph (WSnnG) from the co-association matrix. They use an approach to defining similarity introduced by Jarvis and Patrick in [58], and extended in [59]. Nodes of the WSnnG correspond to objects and edges are weighted according to the shared nearest neighbor similarity. Nodes are also weighted according to a measure derived from the shared nearest neighbors, and node weights are used to determine cluster sizes. The WSnnG is then partitioned using the METIS algorithm [55, 56]. In [57], an approach to pruning the WSnnG is presented.

Ayad, Basir, and Kamel [27] develop a method that uses a co-association matrix. The rows of the co-association matrix are normalized to represent association distributions for the objects. For a given number of clusters k, k cluster prototypes are selected from the set of association distributions based on entropy maximization of the selected prototypes and maximization of the generalized Jensen Shannon (JS) divergence among the selected prototypes. These distributions are then grouped by minimizing their JS divergences to the selected prototypes. By aggregating the grouped distributions (by averaging), empirical cluster conditional probability distributions are computed, and objects are assigned to their most probable clusters.

Fern and Brodley [35] aggregate an ensemble of soft partitions generated by applying the EM algorithm [60, 61] on random projections [62–65] of the data. For each data object, EM generates the soft clustering $p(c_l^i|x_j, \theta)$, for $l = 1, \ldots, k$, representing the probability that object x_j belongs to each cluster under the model θ of a mixture of k Gaussians in the projected space. Corresponding to the i-th soft clustering, a similarity matrix $\mathbf{P}^i$ between pairs of objects is computed, where each entry represents the probability that a pair of objects x_j and x_g belongs to the same cluster under model θ, which is calculated as $p_{jg}^i = \sum_{l=1}^{k} p(c_l^i|x_j, \theta)p(c_l^i|x_g, \theta)$. In order to aggregate an ensemble, the values of p_{jg}^i are averaged, resulting in an aggregated similarity

matrix **P** estimating the probability that each pair of objects belongs to the same cluster. The complete-link agglomerative clustering algorithm is then applied to obtain a k-cluster partition.

Lange and Buhmann [37] present an approach based on a non-negative matrix factorization [52] of the aggregated (and normalized) co-association matrix representing the joint probability $p(x_j, x_g)$ of observing two objects in the same cluster. The matrix is factorized to obtain estimates for class-posteriors and class-likelihoods. The EM algorithm [60] is used to optimize the log-likelihood of the model referred to as Probabilistic Label Aggregation.

Hypergraph and Meta-graph Structures

Strehl and Ghosh [29] present heuristic partitioning algorithm for two types of similarity-based structures. The first is a hypergraph where nodes represent the data objects and hyperedges connect the member of a cluster. The second is meta-graph where clusters of objects are viewed as the nodes of an undirected graph (or meta-graph) and edges are weighted using the binary Jaccard measure, as the ratio of the intersection to the union of each pair of clusters.

It is noted that in [29], the mutual information is defined as a measure of "agreement" (or similarity) between partitions. Specifically, the mutual information measures the statistical information shared between two distributions, where partitions are represented as probability distributions of corresponding categorical variables. An average normalized mutual information (ANMI) criterion was introduced to measure the average amount of statistical information shared between a partition and the ensemble. The optimal consensus partition is defined as the partition that maximizes the ANMI criterion.

Let C^i and C^r denote the random variables describing the partitions π^i and π^r, consisting of k_i and k_r clusters, respectively. The distribution $p(c^i)$ is given by $p(c_l^i) = \frac{n_l^i}{n}$ for $l = \{1, \ldots, k_i\}$, where n_l^i denotes the number of objects assigned to cluster c_l^i according to partition π^i. Let $I(C^i, C^r)$ denote the mutual information between C^i and C^r, and $H(C^i)$ denote the Shannon entropy of C^i. $I(C^i, C^r)$ is a metric but has no upper bound. To obtain a measure with a 0 to 1

range, a normalized mutual information NMI measure is defined in [29] as,

$$\text{NMI}(C^i, C^r) = \frac{I(C^i, C^r)}{\sqrt{H(C^i)H(C^r)}} \tag{2.3}$$

The ANMI criterion is given by,

$$\text{ANMI} = \frac{1}{b}\sum_{i=1}^{b} \text{NMI}(C, C^i) \tag{2.4}$$

where C is the random variable associated with any partition of $\mathcal{X}$, $\pi \in \Pi$. The optimal consensus partition is that which maximizes the ANMI criterion.

To extract the consensus partition, Strehl and Ghosh [29] propose three heuristic algorithms based on the graph-partitioning algorithms in [55,56]. Unlike the CSPA algorithm described earlier, a Hyper Graph Partitioning Algorithm (HGPA), and Meta CLustering Algorithm (MCLA) represent computationally more efficient alternatives. It is noted that these algorithms do not explicitly evaluate the ANMI criterion but are applied as effective heuristic algorithms.

In the case of the HGPA algorithm, the maximum mutual information objective is approximated by with a constrained minimum cut objective. The cluster ensemble problem is viewed as a hypergraph partitioning problem where hyperedges represent clusters and the objective is to cut a minimal number of hyperedges for obtaining k unconnected components of approximately the same size. Hence, the approach is suited for data with balanced cluster sizes.

In case of the MCLA algorithm, a meta-graph is constructed and partitioned, resulting in a clustering of clusters. Each object is then assigned to its most associated meta-cluster.

2.2.3 Consensus Based On A Categorical Feature-Space

Topchy, Jain and Punch [38,41] present a probabilistic model of consensus using a finite mixture of multinomial distributions in a space of clusterings, viewed as categorical variables. The cluster labels $\mathbf{y}_j$ for the object x_j are modeled as random variables drawn from a probability distribution representing a mixture of multi-variate components. A maximum likelihood estimation problem

is formulated, where the most fitting mixture density is obtained by maximizing the likelihood function. In this approach, the EM algorithm [60,61] is applied, leading to a consensus partition.

In another approach, Topchy et al. [30, 38] point out that the objective function used by Strehl and Ghosh [29] is based on the classical Shannon definition of mutual information. On the other hand, by considering another information-theoretic definition of entropy, the mutual information criterion becomes equivalent to the category utility function introduced by Gluck and Corter [66]. In the context of partition ensembles, the category utility function $U(\pi, \pi_i)$ measures the agreement between two partitions as the difference between the expected number of labels of partition π^i that can be correctly predicted with the knowledge of π and without it [38]. It is given by,

$$U(\pi, \pi_i) = \sum_{q=1}^{k} p(c_q) \sum_{l=1}^{k_i} p(c_l^i|c_q)^2 - \sum_{l=1}^{k_i} p(c_l^i) \tag{2.5}$$

The overall utility of a partition with respect to the ensemble partitions is defined as the sum of the pairwise utilities, given by,

$$U(\pi, \mathcal{P}) = \sum_{i=1}^{b} U(\pi, \pi^i). \tag{2.6}$$

By considering the generalized entropy $H^s(C)$, where $\lim_{s\to 1} H^s(C) = -\sum_{q=1}^{k} p(c_q) \log p(c_q)$, the generalized mutual information can be defined [38], where, in particular, the quadratic mutual information $I^2(\pi, \pi^i)$ becomes,

$$I^2(\pi, \pi^i) = 2U(\pi, \pi^i). \tag{2.7}$$

Based on Mirkin's proof in [67], maximizing the partition utility defined in Eq. 2.6 is equivalent to minimizing the squared error clustering criterion, for a fixed number of clusters k in π. Hence, the minimum quadratic mutual information criterion is also equivalent to the classical intra-cluster minimum variance. So, the categorical variables representing the partition are standardized

in [30,38], to transform them to quantitative variables, and a minimum squared error clustering algorithm (the k-means) is applied to extract the consensus partition. The consensus algorithm is referred to as the quadratic mutual information algorithm (QMI).

2.2.4 Consensus Via Voting-Based Aggregation

In seeking to establish a parallel approach to the aggregation of supervised classifiers [17,18,21], a distinct class of consensus methods for partitions, referred to as voting-based methods, was developed as proposed in [28,31,33,39,42,43]. Unlike other consensus methods described earlier, voting-based methods do not avoid the so-called cluster label correspondence problem. Instead, they search for an optimal relabeling and aggregation of the ensemble partitions. Typically, an aggregated partition is computed by averaging the relabeled ensemble partitions with respect to a representative partition, referred to as a reference partition.

An important element of voting-based aggregation is the optimal relabeling of an ensemble partition with respect to a representative partition of the ensemble. This element defines what we refer to as the *pairwise relabeling* or the *voting* problem. The problem is commonly viewed as a weighted bipartite matching problem, where one looks for an optimal cluster label permutation for each ensemble partition to maximize its labeling agreement with respect to a representative labeling [28,31,33,42,43].

Dimitriadou, Weingessel, and Hornik [33] note that solving the voting-based aggregation problem requires the simultaneous optimization of the partitions relabeling with respect to the representative (aggregated) partition and of finding the aggregated partition that optimally represent the ensemble partitions. Hornik [34] further notes that finding the optimally permuted partitions represents a *multi-dimensional assignment problem* (MAP), which unlike the bipartite matching problem that corresponds to a linear sum assignment problem (LSAP), is NP-hard, with branch-and-bound approaches being computationally infeasible for typical ensemble sizes ($b \geq 20$).

Dimitriadou et al. [33] consider fuzzy ensembles and present a theoretical derivation of an

efficient algorithm that iteratively finds the bipartite matching solution which minimizes the mean squared error of the relabeled partition with respect to a representative partition. The algorithm was applied on partitions with a fixed number of clusters in [33] and is referred to as "Voting". We refer to it here as "bVote" in reference to bipartite matching as the underlying formulation of the voting problem. For partitions with a variable number of clusters, empty (dummy) clusters are added to the partition with fewer clusters in order to compute the bipartite matching solution. After the aggregated partition is computed, if it is required to obtain a smaller number of clusters, "neighboring" clusters may be merged based on a similarity measure, so as to get the desired number of clusters [33]. The voting-merging algorithm is referred to as VMA [53].

Gordon and Vichi [42] presented methods for fitting a fuzzy consensus partition to a set of hard or fuzzy partitions. A weighted least-squares objective function is used, where weights allow the ensemble partitions to have different degrees of importance. Classes are permuted so as to establish the correspondence between the classes of the profile, and those of the consensus partition, by solving a cost assignment problem. As in [33], to match two partitions with different numbers of clusters, dummy clusters are introduced into the matching cost matrix so that a binary $k_i \times k_i$ permutation matrix is computed by solving the corresponding assignment problem. To aggregate an ensemble, an algorithm that iterates between two steps is described. In one step, optimal relabelings for the ensemble partitions are determined by optimally matching their cluster labels with the current reference partition, where the initial reference is randomly generated, and in the second step, the reference partition is updated as the (weighted) average of the current relabeled ensemble partitions.

Dudoit and Fridlyand [31] present a bagging procedure ("BaggClust1") in which the bipartite matching solution is computed for each bootstrap clustering with respect to a pre-clustering of the objects. It is assumed the ensemble partitions have a fixed number of clusters which is equal to the number of clusters of the original clustering representing the reference partition. The number of clusters is also equal to the desired number of consensus clusters. The consensus partition is obtained by assigning a bagged cluster label for each object by plurality voting (i.e., by taking

the majority cluster label for each object). Furthermore, the proportions of votes in favor of the bagged cluster labels reflect the confidence of cluster assignments for the objects.

Fischer and Buhmann [28] present a voting-based aggregation algorithm similar to the iterative algorithm in [33]. It is applied in [28] on hard ensembles based on bootstrap resampling. It is assumed that all ensemble partitions have the same number of clusters, which is equal to the target number of clusters in the consensus clustering. At each iteration, the best cluster label permutation is determined by solving a corresponding weighted bipartite matching problem, maximizing the empirical cluster assignment probabilities estimated from the previous relabeling. The final consensus partitions is determined based on a maximum likelihood mapping function. In [28], it is argued that the iterative algorithm is better that the fixed-reference aggregation algorithm "BaggClust1" [31], as the latter would be sensitive to a poor reference partition.

In [39], we introduced the idea of cumulative voting as a new solution for the relabeling problem of a given clustering with k_i clusters with respect to a reference clustering with k_0 clusters, where k_i and k_0 may be unequal. Cumulative voting is a type of rated voting that is related to ranked types of voting such as the Borda count. Instead of binary one-to-one votes, numeric values (ratings) are computed for each option (i.e. reference cluster), such that they sum up to a pre-specified total. Note that the term "cumulative" refers to the property that the computed vote weights for each voting cluster (i.e., each cluster of an ensemble partition) must add up to a pre-specified value. An un-normalized cumulative voting was developed scheme where ratings must sum up to the size of the voting cluster. Based on viewing each partition as a categorical variable, we also developed a normalized cumulative voting scheme was also developed, where votes are weighted according to the conditional probability of each reference cluster, given a cluster of an ensemble partition. In this case, the vote weights must sum to 1. A fixed-reference aggregation algorithm was applied and a new adaptive algorithm was also proposed for aggregating an ensemble in a particular order according to a proposed criterion that maximizes the average amount of information. Furthermore, the problem of extracting an optimal consensus partition with a fewer number of clusters than the ensemble partitions is formulated

as the problem of finding a compressed summary of the estimated distributional representation of the ensemble that preserves maximum relevant information [39, 68]. Based on the information bottleneck formulation of Tishby, Pereira and Bialek [8], an efficient agglomerative algorithm which minimizes the Jensen-Shannon divergence within the cluster is proposed for estimating a compressed consensus partition.

2.3 Ensemble Generation Techniques

The nature and optimality of the consensus solution for a partition ensemble depend not only on the devised combination method, but also on the ensemble generation mechanism which produces the ensemble partitions. In fact, the thesis demonstrates for voting-based methods, that the quality of obtained consensus solution highly depends on the suitability of the consensus method to combining the type of generated ensemble. Several ensemble generation techniques for partition ensembles have been considered in recent literature as described below.

Strehl and Ghosh [29] considered selecting a *portfolio* of clustering methods, where a number of different methods in conjunction with different distance/similarity measures are applied to generate multiple clustering solutions for a given dataset. Furthermore, they considered *object-distributed* and *feature-distributed* ensembles as a means for addressing distributed data mining problems. Object-distributed ensembles address the problem of clustering data consisting of either a very large number of objects or of data-subsets that are stored in different sites with possible privacy constraints, and where a global clustering solution is desired. On the other hand, feature-distributed ensembles address the problem of clustering data characterized by very high dimensional feature spaces. Random feature subspaces are generated and the data is partitioned in each subspace. It is noted that, in the case of supervised learning, a random subspace method [20] was investigated for improving classifiers' accuracy by constructing decision forests in randomly chosen subspaces. Feature-distributed ensembles can be useful for dealing with very high dimensional datasets by representing the data in multiple spaces of reduced dimensionality.

Fern and Brodley [35] generate partition ensembles based on random projections of the data. Random projections correspond to a transformation method that can improve the clustering quality for high dimensional data [62–65]. High dimensionality represents a challenging issue in data clustering. It leads to a problem known as the curse of dimensionality where the vectorial representation of the data becomes too sparse, leading to the inability to find structure in the data. Furthermore, the presence of irrelevant and noisy features can lead to misleading clustering solutions. The application of random projections in [35], in conjunction with the EM clustering (of Gaussian mixtures) is motivated by Dasgupta's result [69], which shows that random projections can change the shape of highly eccentric clusters to be more spherical.

Bootstrap resampling of the original data represents another fundamental ensemble generation technique, where each bootstrap sample of the data is partitioned to produce an ensemble of different partitions. The approach was considered in several studies such as Leisch [70], Dudoit and Fridlyand [31], Fischer and Buhmann [28], Minaei-Bidgoli, Topchy, and Punch [71], and Ayad and Kamel [72]. It represents a scheme similar to bagging [17] for combining predictors. Resampling methods are well established approaches for obtaining accurate estimates of data statistics [71]. In particular, bootstrap resampling is a general method of sampling with replacement that was shown to improve the accuracy and reliability of predictions, which are typically combined by averaging (when predicting a numerical outcome) or via plurality voting (when predicting a categorical outcome). Monti et al. [40] also considered bootstrap resampling in conjunction with multiple random restarts of clustering algorithms (such as k-means, model-based Bayesian clustering, or SOM). They also considered ensemble based on random restarts of these algorithms in conjunction with gene resampling, for gene expression data.

Since clustering algorithms are typically devised as randomized search algorithms, the resulting clustering solutions depend on the applied randomization, such as the selection of the random initial seeds for k-means algorithm. This dependency on random restarts motivates another type of ensemble generation techniques where each ensemble partition corresponds to a solution for a random restart. Fred and Jain [24] considered this ensemble generation technique in conjunc-

tion with the k-means algorithm to improve the stability of clustering solutions. Dimitriadou et al. [33] and Gordon et al. [42] apply several runs of a fuzzy clustering algorithm such as the the fuzzy c-means (FCM) to generate the ensemble partitions.

The approach in [24], based on the k-means algorithm, is further extended by generating an *overproduced numbers of clusters* [25,36]. That is, each ensemble partition consists of a relatively large number of clusters compared to the desired or suspected number of true clusters. This overproduction of clusters creates a split-and-merge scheme where the ensemble consists of fine-resolution partitions whereas the consensus partition is a coarser partition reflecting the global cluster structure of the data. To induce more diversity among the ensemble partitions, the number of overproduced clusters is randomly selected for each ensemble partition in [36, 73].

Topchy, Jain, and Punch [30] proposed the generation of *weak* partitions using two different techniques. In the first, the partitions are generated based on random one-dimensional projections of the original feature space. In the second technique, partitions are generated by splitting the data using a number of random hyperplanes. That is, the random hyperplanes dissect the d-dimensional space of the data, and objects separated by the hyperplanes are assigned to different clusters. When only one hyperplane is used, the data is splitted into two clusters. The idea of weak partitions is to seek the combination of simple and cheaply computed partitions rather than complex ones. It is noted that, in the case of supervised learning, weak classifiers were considered in the *boosting* method introduced in [18], where substantial gains in accuracy were achieved.

2.4 Analysis of Consensus Partitions

A substantial amount of theoretical analysis was developed in the early work on consensus of classification models in the areas of discrete mathematics, theoretical computer science [15,48,48], and pattern recognition [46]. For instance, Barthelemy and Leclerc [15] studied the properties of the median procedure for partitions and compared it with axiomatic and constructive approaches. The median partition problem was shown to be NP-Complete [15], and heuristic algorithms for

finding approximate solutions were studied. Régnier [48] studied the convergence properties of central partitions. Neumann and Norton [46] show that an appropriate consensus partition of a given profile is not a single partition but is one that should lie in a *consensus interval*, where the extremes of this interval are characterized axiomatically. Thus, they show that any reasonable *consensus function* must take its values in this interval.

Recent consensus methodologies are typically validated empirically. The accuracy and stability of the obtained consensus partitions are evaluted on artificial and real datasets and compared with the individual partitions of the ensemble or with other consensus algorithms. Kuncheva and Vetrov [73] study the empirical relationship between the stability and accuracy of consensus partitions with respect to the number of clusters, for the k-means algorithm. They observe that the relationship highly depends on the data set, with the correlation varying from nearly +1 to nearly -1. They introduce a combined stability index as the sum of the pairwise individual and ensemble stabilities. The correlation of the new index with the ensemble accuracy was found to be more consistent, and was used to determine the number of clusters. In [74], Hadjitodorov, Kuncheva, and Todorova study the relation between the diversity of the ensemble and quality of the consensus partition. They found that ensembles with a moderate level of diversity lead to more accurate consensus partitions. Building upon this finding, a procedure for generating and selecting ensembles with median diversity is presented.

In [43], Topchy et al., consider the theoretical validation of cluster ensembles and present a formal analysis of the convergence properties of consensus partitions. The analysis considers two approaches to the consensus problem. The first is based on a stochastic partition generation model and a voting-based consensus method. In the voting-based method, the pairwise relabeling problem is formulated as a bipartite matching problem and solved using Kuhn's Hungarian method. Then, the plurality voting rule is applied to determine the consensus partition. The number of clusters is assumed to be fixed. Further details of this approach are given in Ch. 3. The second approach considers the properties of the mean partition with respect to a metric on the space of partitions. In both cases, the consensus solution is shown to converge to the under-

lying generating clustering (which is assumed to represent the true clustering), as the number of partitions in the ensemble increases, assuming that each partition gives a better than random clustering compared to the underlying clustering.

Chapter 3

Voting-Based Partition Aggregation

In this chapter, a new framework for the voting problem is defined in Sec. 3.1. The voting-based aggregation problem and a general iterative algorithm are described in Sec. 3.2. A simulation-based analysis is presented in Sec. 3.3, for comparing the cumulative voting and bipartite matching schemes. Finally, possible interpretations of the aggregated partition are discussed in Sec. 3.4.

3.1 A New Framework for the Voting Problem

Recall from Ch. 1 that voting-based consensus methods [28, 31, 33, 39, 42, 43] derive an ensemble representation consisting of a central aggregated partition by seeking an optimal relabeling of the ensemble partitions. In general, the optimal relabeling of the ensemble partitions is addressed through a pairwise relabeling of each ensemble partition with respect to a representative partition, where the pairwise relabeling problem is also referred to here as the voting problem. Each cluster of a given ensemble partition is viewed as a "voter" that votes for the representative clusters according to a defined voting .

Let $\mathcal{X}$ denote a set of n data objects, and let a partition of $\mathcal{X}$ into k clusters be represented by

an $n \times k$ stochastic matrix $\mathbf{U}$, with a row for each object, and a column for each cluster, such that $\sum_{q=1}^{k} u_{jq} = 1, \forall j$. In general, $\mathbf{U}$ may represent a hard partition with $u_{jq} \in \{0, 1\}$ or a soft partition with $u_{jq} \in [0, 1]$. To obtain a hard from a soft partition, u_{jr} is set to 1, for $r = \arg\max_q u_{jq}$, and $u_{jq} = 0$ otherwise. A hard partition is also represented by a labeling n-vector $\mathbf{y}$.

Let $\mathcal{U} = \{\mathbf{U}^i\}_{i=1}^{b}$ denote an ensemble of partitions. The voting-based aggregation problem is concerned with searching for an optimal relabeling of the ensemble partition and for a central aggregated partition denoted as $\bar{\mathbf{U}}$ that summarizes the ensemble partitions.

Note that, in principle, the proposed cumulative voting scheme and the bipartite matching scheme are applicable to both hard and soft ensembles. In fact, the bipartite matching scheme has already been studied for hard ensembles in [28,31,43] and for soft ensembles in [33,42]. Basic modifications required for applying cumulative voting to soft ensembles are described in [39]. However, for simplicity and a focused analysis, only hard ensembles are considered here as input.

The aggregated partition is a soft partition, as computed using either the cumulative voting or bipartite matching schemes. Note that "soft" is a general term that is used in the literature either to describe a partition obtained using a statistical model-based clustering algorithm that maximizes the likelihood function [75,76], where u_{jq} reflects the uncertainty about the associated classification of each data object, or to describe a partition obtained using a clustering algorithm that optimizes a fuzzy objective function [77,78], where u_{jq} reflects a fuzzy membership value. In this thesis, the aggregated partition is viewed as a soft partition in a statistical sense. Specifically, it is obtained via least-squares estimation.

In this section, a general formulation of the voting problem is proposed in Sec. 3.1.1. The cumulative voting and bipartite matching formulations are described in Sec. 3.1.2 and 3.1.3, respectively. In Sec. 3.1.4, an illustrative example is given to highlight the characteristics of the two different voting formulations.

3.1.1 General Formulation

Voting is viewed as the problem of estimating the assignments of the objects to representative clusters $\mathcal{C}^0 = \{c_1^0, \ldots, c_{k_0}^0\}$, given their assignments to the clusters of an ensemble partition $\mathcal{C}^i = \{c_1^i, \ldots, c_{k_i}^i\}$, such that the estimation errors compared to the representative partition $\mathbf{U}^0$ are minimized. Let the random vectors $C^i = (C_1^i \ldots C_{k_i}^i)$ and $C^0 = (C_1^0 \ldots C_{k_0}^0)$ denote the clusters of $\mathbf{U}^i$ and $\mathbf{U}^0$, respectively, where each variable C_q^0 is considered to take real values in $[0, 1]$, such that $\sum_{q=1}^{k_0} u_{jq}^0 = 1$. That is, $\mathbf{U}^0$ is a soft partition. The voting problem can be viewed as seeking a function of $\mathbf{U}^i$, $\vartheta^i(\mathbf{U}^i)$, that establishes a relationship between $\{C_l^i\}_{l=1}^{k_i}$ and $\{C_q^0\}_{q=1}^{k_0}$, such that a loss function $L^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i))$ is minimized, where L^i is referred to here as the voting (or pairwise relabeling) loss. The function $L^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i))$ penalizes the errors in the estimated values of $\vartheta^i(\mathbf{U}^i)$ compared to $\mathbf{U}^0$. The problem of finding $\vartheta^i(\mathbf{U}^i)$ is given by Eq. 3.1.

$$\min_{\vartheta^i(\mathbf{U}^i)} L^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i)). \tag{3.1}$$

The above formulation of the voting problem is equivalent to a *regression problem* (supervised learning with numerical output variable(s)). Specifically, it corresponds to a multiple regression problem with multiple output (response) variables $\{C_q^0\}_{q=1}^{k_0}$ and multiple input (predictor) variables $\{C_l^i\}_{l=1}^{k_i}$. Based on this formulation, the regression function $\vartheta^i(\mathbf{U}^i)$, which may be referred to as the voting (or relabeling) function, estimates the conditional expectation of C^0 given C^i, $\mathrm{E}(C^0|C^i)$, and is a vector function, $\vartheta^i(\mathbf{U}^i) = (\vartheta_1^i(\mathbf{U}^i), ..., \vartheta_{k_0}^i(\mathbf{U}^i))$ [10].

Let $\mathcal{L}^i$ denotes the i-th *learning set* corresponding to the partition pair $\mathbf{U}^i$ and $\mathbf{U}^0$, and consisting of the vectors $\{(\mathbf{u}_j^i, \mathbf{u}_j^0)\}_{j=1}^n$, where $\mathbf{u}_j^i$ is the jth row vector of $\mathbf{U}^i$ and represents a k_i input vector, and $\mathbf{u}_j^0$ is the j-th row vector of $\mathbf{U}^0$ and represents a k_0 output "target" vector. The goal is to use $\mathcal{L}^i$ to estimate $\vartheta^i(\mathbf{U}^i)$. In the voting problem, it is only the learning but not the generalizing (prediction) aspect of regression that is applied.

The proposed framework generalizes the bipartite matching scheme, which establishes binary one-to-one relationships between two sets of clusters, by exploring the idea of a soft relabeling (or

soft voting) and addressing the problem of establishing real-valued many-to-many relationships between the clusters of a given partition and the clusters of a representative partition. An important feature of the proposed framework is its added flexibility, which makes suitable for modeling complex relations arising from substantial variability between partitions, such as a variable number of clusters. An illustrative example is given in Sec. 3.1.4. Note that in the case of bipartite matching, empty (dummy) clusters need to be added to the partition with a smaller number of clusters as proposed in [33, 42], or the ensemble must be constrained to partitions with a fixed number of clusters that is also equal to the desired number of consensus clusters as in [28, 31, 43].

It is noted that, in a regression problem, the form of the function $\vartheta^i(\mathbf{U}^i)$, that underlies the relationship between the input and output variables, is generally unknown [10]. Below, the regression model underlying the cumulative voting scheme introduced in [39] is derived.

3.1.2 Cumulative Voting

In [39], two types of cumulative voting are investigated; the normalized and un-normalized schemes. The term "cumulative" refers to the property that the computed vote weights for each variable C_q^0 must add up to a pre-specified value. In the case of the normalized scheme, the sum must be 1, and in the case of the un-normalized scheme, the sum must be equal to the size of the voting cluster. In this chapter, the focus is on the normalized scheme. More details on the un-normalized scheme are given in Ch. 4, where its close relationship with the co-association matrix representation for partition ensembles is outlined. Whenever the term "cumulative voting" is used in the thesis, it refers to the normalized scheme.

The cumulative voting scheme represents a simple but reasonable regression method. It corresponds to fitting a linear model by least squares. It is considered reasonable for its minimal assumptions about the underlying model of the data. Assuming a linear model for each output variable, $\vartheta^i(\mathbf{U}^i)$ is written in matrix notation as,

$$\vartheta^i(\mathbf{U}^i) = \mathbf{U}^i\mathbf{W}^i, \tag{3.2}$$

where $\mathbf{W}^i$ is a $k_i \times k_0$ matrix of coefficients denoted as w^i_{lq}.

Let $\mathbf{V}^i = \vartheta^i(\mathbf{U}^i)$, which is a $n \times k_0$ matrix. To fit the linear model in Eq. 3.2 to the learning set $\mathcal{L}^i$, the coefficients $\mathbf{W}^i$ are estimated to minimize the mean squared errors, as given in Eq. 3.3:

$$L^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i)) = \text{MSE}^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i)) = \frac{1}{n}\sum_{j=1}^{n}\sum_{q=1}^{k_0}\left(u^0_{jq} - v^i_{jq}\right)^2, \tag{3.3}$$

which is written in matrix notation as follows,

$$\text{MSE}^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i)) = \frac{1}{n}\text{tr}[\left(\mathbf{U}^0 - \mathbf{U}^i\mathbf{W}^i\right)^T\left(\mathbf{U}^0 - \mathbf{U}^i\mathbf{W}^i\right)]. \tag{3.4}$$

The solution is obtained by differentiating with respect to $\mathbf{W}^i$ and is given by Eq. 3.5. The estimated partition $\hat{\mathbf{V}}^i$ is given by $\hat{\mathbf{V}}^i = \mathbf{U}^i\hat{\mathbf{W}}^i$.

$$\hat{\mathbf{W}}^i = \left({\mathbf{U}^i}^T\mathbf{U}^i\right)^{-1}{\mathbf{U}^i}^T\mathbf{U}^0. \tag{3.5}$$

It is easy to see that the normalized scheme in [39] corresponds to the linear model with least squares fit, as defined above in Eqs. 3.2 and 3.3 by noting that, for hard ensemble partitions, the term $({\mathbf{U}^i}^T\mathbf{U}^i)$ in Eq. 3.5 is a diagonal matrix, and hence Eq. 3.5 gives the same expression for the coefficients as computed in [39], which is given by,

$$\hat{w}^i_{lq} = \frac{1}{n^i_{l.}}\sum_{j\in\{1,\ldots,n\}\,:\,u^i_{jl}=1} u^0_{jq}, \tag{3.6}$$

where $n^i_{l.}$ denotes the number of objects assigned to cluster c^i_l, and $\hat{v}^i_{jq} = \hat{w}^i_{lq}$ if $u^i_{jl} = 1$, and 0 otherwise. If $\mathbf{U}^0$ is also a hard partition, then $\hat{w}^i_{lq} = \frac{n^i_{lq}}{n^i_{l.}}$, where n^i_{lq} is the number of objects assigned to clusters c^i_l and c_q. Note that one would have a hard reference partition if the applied

aggregation algorithm follows a fixed-reference approach, whereby an initial reference partition is used as a common representative partition for all the ensemble partitions and remains unchanged throughout the aggregation process. In this thesis, a fixed-reference approach is not considered. Instead, an iterative aggregation approach is adopted.

Based on the constraint on the output variables $\{C_q^0\}_{q=1}^{k_0}$, the estimated values $\hat{v}_{jq}^i$ must sum to 1, $\sum_{q=1}^{k_0} \hat{v}_{jq}^i = 1$. Note that one may consider C^0 as a categorical variable with k_0 categories, $\{c_1^0, \ldots, c_{k_0}^0\}$. Based on the squared error loss estimation, the estimate $\hat{\mathbf{V}}^i$ of the conditional expectation can be viewed as an estimate of the posterior probability $\mathrm{E}(C^0|C^i) = \Pr(C^0|C^i)$ [10]. Classifying to the most probable class $\hat{\mathbf{V}}^i = \arg\max_{c_q^0 \in C^0} \Pr(c_q^0|C^i)$ gives the *Bayes classifier*, with the *Bayes rate* as the error rate [10]. Hence, this rate gives a lower bound on the achievable error rate for a relabeling $\vartheta^i(\mathbf{U}^i)$, based on least squares loss. The error rate is denoted here as $\mathrm{Err}^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i))$, which can be computed by considering C^0 as a categorical variable. That is, it is computed in the special case where the voting problem is viewed as a classification problem.

3.1.3 Bipartite Matching

A combinatorial optimization problem with a constrained least squares objective underlies the bipartite matching scheme as described in [33, 42] and as given below in Eq. 3.7. It is not a regression problem, but can be expressed in the same framework as follows. Suppose that $k_i = k_0 = k$, otherwise, dummy clusters may be added to the partition with the lower number of clusters. Let the $\mathbf{V}^i = \vartheta^i(\mathbf{U}^i)$ be defined as a column (cluster) permutation of partition $\mathbf{U}^i$, which can be written as $\mathbf{V}^i = \mathbf{U}^i\mathbf{W}^i$, where $\mathbf{W}^i$ is a $k \times k$ permutation matrix. The problem is to find $\mathbf{W}^i$ that minimizes $\mathrm{MSE}^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i))$ subject to constraints on w_{lq}^i as defined in Eq. 3.7. It is also known as a *linear sum assignment problem*, LSAP, or a weighted bipartite matching problem [34].

$$\begin{aligned} &\min_{w_{lq}^i} \frac{1}{n} \sum_{j=1}^{n} \sum_{q=1}^{k} \left(u_{jq}^0 - v_{jq}^i\right)^2 \\ &\text{Subject to } \sum_{l=1}^{k} w_{lq}^i = \sum_{q=1}^{k} w_{lq}^i = 1, \text{ where } w_{lq}^i = 0 \text{ or } 1. \end{aligned} \tag{3.7}$$

The optimal solution for the problem in Eq. 3.7 is obtained using Kuhn's Hungarian method [33]. Since the bipartite matching scheme minimizes the same loss function as cumulative voting, but as a consequence of the additional constraints as given in Eq. 3.7, the MSE^i achievable using the bipartite matching is bounded from below by the MSE^i achievable using cumulative voting.

The minimization problem in Eq. 3.7 is equivalent to the constrained maximization of $\text{tr}(\mathbf{G}^{iT}\mathbf{W}^i)$ [33], where $\mathbf{G}^i$ is the contingency matrix of $\mathbf{U}^i$ and $\mathbf{U}^0$, $\mathbf{G}^i = \mathbf{U}^{iT}\mathbf{U}^0$. Unlike cumulative voting, the estimated partition $\mathbf{V}^i$ is a hard partition, when $\mathbf{U}^i$ is hard. In the case of hard ensemble partitions, the problem is equivalent to minimizing the probability of error p_e^i subject to the constraints defined in Eq. 3.7, where p_e^i is given by,

$$p_e^i = \frac{1}{n} \sum_{l=1}^{k} \sum_{q=1}^{k} g_{lq}^i (1 - w_{lq}^i). \tag{3.8}$$

If both $\mathbf{U}^i$ and $\mathbf{U}^0$ are hard partitions, the constrained error rate $\text{Err}^i(\mathbf{U}^0, \vartheta^i(\mathbf{U}^i))$ is minimized, which is given by,

$$\text{Err}^i = \frac{1}{n} \sum_{l=1}^{k} \sum_{q=1}^{k} n_{lq}^i (1 - w_{lq}^i), \tag{3.9}$$

where n_{lq}^i is the number of objects assigned to clusters c_l^i and c_q^0.

Again, due to the additional constraints, the error rate achievable using the bipartite matching is bounded from below by the error rate achievable using cumulative voting when the latter is followed by classifying to the most probable class.

3.1.4 Illustrative Example

Consider a set of ten data objects $\mathcal{X} = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\}$. Suppose a reference partition $\mathbf{U}^0$ and an ensemble partition $\mathbf{U}^i$ are given as follows, where $\mathbf{U}^0$ partitions the objects into $k_0 = 5$ clusters, whereas $\mathbf{U}^i$ partitions them into $k_i = 2$ clusters.

$$\mathbf{U}^0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \text{ and } \mathbf{U}^i = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

Relabeling $\mathbf{U}^i$ based on cumulative voting gives a coefficient matrix $\mathbf{W}^i$ and a relabeled partition $\mathbf{V}^i$ as follows:

$$\mathbf{W}^i = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.3333 & 0.3333 & 0.3333 \end{bmatrix}, \text{ and } \mathbf{V}^i = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0.3333 & 0.3333 & 0.3333 \end{bmatrix}.$$

That is, in the case of cumulative voting, the uncertainties associated with assigning the objects belonging to cluster 1 of $\mathbf{U}^i$ to each of the five clusters of $\mathbf{U}^0$ are given by first four rows

of $\mathbf{V}^i$. This soft assignment reflects the fact that the objects belonging to cluster 1 of $\mathbf{U}^i$ are divided equally among the first two clusters of $\mathbf{U}^0$. Similarly, equal probabilities of assigning the objects belonging to cluster 2 of $\mathbf{U}^i$ to each of the last three reference clusters are reflected in the last six rows of $\mathbf{V}^i$.

Based on bipartite matching, relabeling $\mathbf{U}^i$ gives a permutation matrix $\mathbf{W}^i$ and a relabeled partition $\mathbf{V}^i$ as follows:

$$\mathbf{W}^i = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \text{ and } \mathbf{V}^i = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

For the bipartite matching scheme, the objects belonging to cluster 1 of $\mathbf{U}^i$ are simply assigned to reference clusters 1, and no objects are assigned to reference cluster 2. Similarly the objects of cluster 2 of $\mathbf{U}^i$ are assigned to reference clusters 3, and no objects are assigned to cluster 4 or 5. The relabeling requires three empty clusters to be created. Unlike the cumulative voting scheme, the assignment of the data objects ignores the fact that each cluster of $\mathbf{U}^i$ is equally divided between more than one reference cluster.

In this thesis, we show evidence that the bipartite matching scheme can be more desirable compared to cumulative voting for a particular type of ensemble. This is demonstrated by the simulation-based analysis of uniform ensembles presented in this chapter. However, the fact that cumulative voting reflects more closely the uncertainty associated with the relabeling makes it more desirable for another type of ensemble, as demonstrated by the empirical evidence presented in Ch. 5.

3.2 Voting-Based Aggregation

In this section, the problem of optimally relabeling and aggregating an ensemble is presented. A general voting-based aggregation algorithm presented in [28, 33] is applied, in conjunction with each of the cumulative voting and bipartite matching schemes.

3.2.1 Formulation

The problem is to estimate a central partition $\bar{\mathbf{U}}$ that minimizes the overall loss $L(\bar{\mathbf{U}}; \{\mathbf{U}^i\}_{i=1}^b) = \frac{1}{b}\sum_{i=1}^b L^i(\bar{\mathbf{U}}, \mathbf{U}^i)$, where $L^i(\bar{\mathbf{U}}, \mathbf{U}^i) = \min_{\vartheta^i(\mathbf{U}^i)} L^i(\bar{\mathbf{U}}, \vartheta^i(\mathbf{U}^i))$, as given by Eq. 3.10.

$$\min_{\bar{\mathbf{U}}} L(\bar{\mathbf{U}}; \mathbf{U}^1, \dots \mathbf{U}^b) = \min_{\bar{\mathbf{U}}} \min_{\vartheta^i(\mathbf{U}^i)} \frac{1}{b}\sum_{i=1}^b \; L^i(\bar{\mathbf{U}}, \vartheta^i(\mathbf{U}^i)). \tag{3.10}$$

Using the least squares objective for the aggregated partition, with respect to the ensemble partitions, the aggregation problem is written as,

$$\min_{\bar{\mathbf{U}}} \mathrm{MSE}(\bar{\mathbf{U}}; \mathbf{U}^1, \dots \mathbf{U}^b) = \min_{\bar{\mathbf{U}}} \min_{\vartheta^i(\mathbf{U}^i)} \frac{1}{b}\sum_{i=1}^b \; \mathrm{MSE}^i(\bar{\mathbf{U}}, \vartheta^i(\mathbf{U}^i)) \tag{3.11}$$

As noted in [33], the aggregation problem in Eq. 3.11 is computationally challenging as it requires the simultaneous optimization of $\vartheta^i(\mathbf{U}^i)$ with respect to $\bar{\mathbf{U}}$ (minimization over the space of $\vartheta^i(\mathbf{U}^i)$) and of the aggregated partition (minimization over the space of $\bar{\mathbf{U}}$). The values of $\vartheta^i(\mathbf{U}^i)$ depend on $\bar{\mathbf{U}}$, and vice versa. For fixed $\vartheta^i(\mathbf{U}^i)$, the optimal $\bar{\mathbf{U}}$ is the soft partition computed as the average $\frac{1}{b}\sum_{i=1}^b \vartheta^i(\mathbf{U}^i)$. Hornik [34] notes that finding the optimally permuted partitions, in a global sense, represents a *multi-dimensional assignment problem* (MAP), which unlike LSAP, is NP-hard, with branch-and-bound approaches being computationally intractable for typical ensemble sizes ($b \geq 20$). An efficient voting-based aggregation algorithm derived in [33] is adopted, as detailed in Sec. 3.2.2.

As a last observation on the formulation of the voting-based aggregation problem, it is noted that one can use the probability of error as the optimization criterion for the cumulative vot-

ing scheme, by classifying to the most likely class, as discussed in Sec. 3.1. In this case, the aggregation problem given in 3.10 is written as follows.

$$\min_{\bar{\mathbf{U}}} p_e(\bar{\mathbf{U}}; \mathbf{U}^1, \dots \mathbf{U}^b) = \min_{\bar{\mathbf{U}}} \min_{\vartheta^i(\mathbf{U}^i)} \frac{1}{b} \sum_{i=1}^{b} \; p_e^i(\bar{\mathbf{U}}, \vartheta^i(\mathbf{U}^i)) \tag{3.12}$$

However, our study focuses on investigating the effect of soft re-labeling of the ensemble partitions and on the optimality and usefulness of the aggregated partition in this case.

3.2.2 Algorithm

Several voting-based aggregation algorithms, which are computationally efficient, are described in recent literature [28, 31, 33, 39, 42, 43]. The simplest type of algorithms follows the approach described in [31, 43], in conjunction with the bipartite matching scheme, and in one of the algorithms in [39], in conjunction with cumulative voting. In this approach, a single common reference partition $\mathbf{U}^0$ is selected. Then, $\{\mathbf{U}^i\}_{i=1}^b$ are optimally re-labeled with respect to $\mathbf{U}^0$. This is followed by computing $\bar{\mathbf{U}}$ by averaging the relabeled partitions. As observed in [28], the drawback of this algorithm is its high dependency on the selected (fixed) reference $\mathbf{U}^0$. However, it is noted that this algorithm represents a suitable approach if the ensemble partitions are known to be uniform. For instance, in the case of the stochastic partition generation model described in [43], where all the ensemble partitions are generated as noisy permutations of an underlying clustering, according to a probability of error, it is shown in that the aggregated partition converges to the underlying clustering, as presented in [43].

In [42], an algorithm that iterates between two steps is described. In one step, $\{\vartheta^i(\mathbf{U}^i)\}_{i=1}^b$ are determined by optimally matching $\{\mathbf{U}^i\}_{i=1}^b$ to the current $\mathbf{U}^0$, where the initial $\mathbf{U}^0$ is randomly generated, and in the second step, $\mathbf{U}^0$ is updated as the (weighted) average of the current $\{\vartheta^i(\mathbf{U}^i)\}_{i=1}^b$. The algorithm converges to at least a local minimum, and as any approximation algorithm, cannot be guaranteed to find global optimal solutions [42]. It is argued, however, that by using a large number of random initializations and keeping the best solution, the chance of

obtaining a global minimum increases, and if the same values of the aggregated partition are repeatedly obtained as the local optimal solution, one can have higher confidence that the global optimal solution has been found [42].

Dimitriadou et al. [33] present a theoretical derivation of another iterative algorithm to find an approximate solution for the aggregation problem in Eq. 3.11. The algorithm works as follows. An initial reference is set as $\mathbf{U}^0 = \mathbf{U}^1$. Then, at each step i, for $i \in \{2, \ldots, b\}$, the locally optimal re-labeling $\vartheta^i(\mathbf{U}^i)$ with respect to the current reference partition is computed, and $\mathbf{U}^0$ is re-computed as the weighted average of the last $\mathbf{U}^0$ and $\vartheta^i(\mathbf{U}^i)$, such that the re-computed reference represents the average of the partitions relabeled thus far (at step i/b). Similar greedy approximation algorithms were also described in [28, 39]. It is noted that for this algorithm, the obtained solution $\bar{\mathbf{U}}$ depends on the ordering of the partitions, and the initial reference $\mathbf{U}^0$. The algorithm can be enhanced by running several passes with random initialization and random order of the partitions, and keeping the best solution [34].

In this chapter, the enhanced iterative algorithm described in Algorithm 1 is applied, as a general voting-based aggregation algorithm, which is referred to as `Vote`. Specific variants of `Vote` based on cumulative and bipartite voting are referred to as `cVote` and `bVote`, respectively. Several passes can be performed by running `Vote` multiple times with random ordering of the ensemble partitions, and keeping the best solution achieved so far (i.e. $\bar{\mathbf{U}}$ with lowest value of MSE($\bar{\mathbf{U}}$,$\mathcal{U}$) is kept). A comparison of the two scheme is presented in Sec. 3.3.

Suppose that $\mathcal{U}$ consists of partitions with a variable number of clusters k_i. When `cVote` is applied, the number of clusters k_0 of $\mathbf{U}^0$ remains fixed throughout the iterations. Thus, $\bar{k} = k_0$, where $\bar{k}$ denotes the number of clusters in $\bar{\mathbf{U}}$. On the other hand, when `bVote` is applied, k_0 can change, and the obtained value of $\bar{k}$ is given by, $\bar{k} = \max_{i=1:b} k_i$, after b iterations.

Algorithm 1 `Vote`

Function $\bar{\mathbf{U}} = \texttt{Vote}(\mathcal{U})$

1: Randomly select a partition $\mathbf{U}^i \in \mathcal{U}$ and assign to $\mathbf{U}^0$
2: **for** $i = 1$ to b **do**
3: For `cVote`, compute $\mathbf{W}^i$ as given by Eq. 3.5.
For `bVote`, compute $\mathbf{W}^i$ by finding the bipartite matching solution to Eq. 3.7.
4: $\mathbf{V}^i = \mathbf{U}^i\mathbf{W}^i$
5: $\mathbf{U}^0 = \frac{i-1}{i}\mathbf{U}^0 + \frac{1}{i}\mathbf{V}^i$
6: **end for**
7: $\bar{\mathbf{U}} = \mathbf{U}^0$.

3.3 Simulation-Based Analysis

The simulation presented in this section illustrates some basic theoretical results and provides a preliminary analysis of the aggregated partition based on each voting scheme, using several partition generation models. The generation models are described in Sec. 3.3.1, and simulation results are presented in Sec. 3.3.2.

3.3.1 Partition Generation Models

As pointed out earlier, a stochastic model for partition generation was considered in [43] for proving the convergence properties of partition ensembles based on the bipartite matching scheme, in conjunction with plurality voting. In this model, ensemble members $\{\mathbf{y}^i\}_{i=1}^b$ are generated as noisy permutations of an underlying labeling $\mathbf{y}^\alpha$ that is considered to represent the true clustering. The model reflects a relatively uniform ensemble where each labeling vector $\mathbf{y}^i$ contains random errors, with probability $p_e^i = p_e \; \forall i$, whereas $\mathbf{y}^i$ is otherwise identical to $\mathbf{y}^\alpha$, modulo cluster label permutation, with $k_i = k_\alpha, \forall i$. When applying the bipartite matching scheme in conjunction with plurality voting, the aggregated partition $\bar{\mathbf{U}}$ was shown to converge to $\mathbf{y}^\alpha$, as the ensemble size increases, and assuming that each ensemble partition gives a better than random clustering result

compared to $\mathbf{y}^\alpha$ [43]. The partition generation model in [43] is simulated and further extended. Furthermore, alternative models for generating non-uniform ensembles are considered.

Generation of Uniform Partitions

The model considered in [43] is based on an underlying cluster structure viewed as the true clustering. Let $\mathbf{y}^\alpha$ be a clustering with k_α clusters which may be generated as a random labeling vector of size n with entries taking value in the set of labels $\{c_1, \ldots, c_{k_\alpha}\}$. To generate an ensemble, b labeling vectors $\{\mathbf{y}^i\}_{i=1}^b$ are generated such that each $\mathbf{y}^i$ is a noisy permutation of $\mathbf{y}^\alpha$. Noise is induced with a probability of random error $p_e^i = p_e \; \forall i$, as follows. For each $i \in \{1, \ldots, b\}$ and each $j \in \{1, \ldots, n\}$, y_j^i is set to be equal to y_j^α, with probability $(1 - p_e)$, or make it acquire an incorrect label $y_j^i \in \{c_1, \ldots, c_{k_\alpha}\}$ such that $y_j^i \neq y_j^\alpha$ with probability p_e, where all $k_\alpha - 1$ incorrect labels are equiprobable, with probability $p_e/(k_\alpha - 1)$ for each label. This is followed by applying a random permutation on each $\mathbf{y}^i$ by drawing a permutation of the labels $\{c_q\}_{q=1}^{k_\alpha}$ from the set of all possible $k_\alpha!$ permutations, with uniform probability.

It is important to note that the convergence of $\bar{\mathbf{U}}$ to $\mathbf{y}^\alpha$ is significant when $\mathbf{y}^\alpha$ represents the true clustering. Otherwise, $\bar{\mathbf{U}}$ can be said to converge to a clustering that is as accurate as $\mathbf{y}^\alpha$ compared to the true clustering. To illustrate this, consider that $\mathbf{y}^\alpha$ represents a noisy labeling compared to some possible true clustering $\mathbf{y}^*$ with probability of error denoted as p_e^α. Then, $\bar{\mathbf{U}}$ can be compared to $\mathbf{y}^*$ by extracting a hard partition from $\bar{\mathbf{U}}$ and measuring the error rate denoted Err^*, for different values of p_e^α. It is noted that, in this model, it is assumed that $k^* = k_\alpha$, whereas in reality, the true number of clusters may be different or unknown.

Generation of Non-Uniform Partitions

As possible alternatives to generating ensembles of uniform partitions, one may consider two different types of non-uniform ensembles. In the first, partitions with a variable number of clusters are generated. In the second, the ensemble partitions have the same number of clusters, but a

variable cluster label distribution. For the first ensemble type, a set of b labelings $\{\mathbf{y}^i\}_{i=1}^b$ with a number of clusters $\{k_i\}_{i=1}^b$ are generated as random n-vectors with entries of $\mathbf{y}^i$ taking value in $\{1,\ldots,k_i\}$, where k_i is randomly selected in a range $[k_{\min}, k_{\max}]$. In the second, a number of clusters $\{k_i\}_{i=1}^b = k$ is assumed, and a set of b labelings $\{\mathbf{y}^i\}_{i=1}^b$ are generated, each as an n-vector with a randomly-generated distribution over the cluster labels $c_l^i \in \{1,\ldots,k\}$, where the probability of each class $p_{c_l^i}$ is generated as follows. Sampling weights $h_l^i \in [h_{\min}, h_{\max}]$ are randomly generated, followed by a normalization to get the class probabilities as $p_{c_l^i} = h_l^i / \sum_{r=1}^k h_r^i$.

In both non-uniform partition generation models, each vector $\mathbf{y}^i$ is re-arranged so that the first n_1^i entries, $y_1^i \ldots y_{n_1^i}^i$, are assigned to the first cluster $c_l^i = 1$, the next n_2^i entries are assigned to the second cluster $c_l^i = 2$, and so on. The ordering of the labeling vectors serves to induce relationships among the ensemble partitions.

In the case of uniform partitions, the variability among the ensemble partitions is due to random errors, whereas in the case of non-uniform ensembles, it is due to other factors such as a variable number of clusters, or a variable class distribution. Unlike uniform partitions, there is no underlying labeling from which the ensemble partitions are directly generated. Therefore, it is not clear how to determine a labeling which can be considered as a possible true clustering, or to assume a true number of clusters. In particular, the evaluation of $\bar{\mathbf{U}}$ compared to possible true clusterings is not as straightforward as in the case of the ensemble generation model in [43].

Based on the introduced voting framework, one computes the uncertainty about the assignment of each data object to a set of clusters described by the random variables $\{C_q^0\}_{q=1}^{k_0}$. These variables are specified based on the initial reference partition. Thus, as a starting point in the simulation-based analysis, it is assumed that the initial reference partition represents the desired labeling $\mathbf{y}^\alpha$, which in turn may represent a noisy replicate of a possible true clustering denoted as $\mathbf{y}^*$. Then, $\bar{\mathbf{U}}$ is compared to $\mathbf{y}^*$ by evaluating Err^* for different values of p_e^α.

However, considering that the initial reference partition is simply selected at random, a better way to deal with this issue is required for a conclusive analysis. A principled information-theoretic approach is approach. Instead of considering the aggregated partition as directly representing

the consensus partition, it can be viewed as an aggregated distributional representation of the ensemble, on the basis of which, an optimal consensus partition may be sought and extracted. This view is further developed in Ch. 5, where the problem of extracting a compressed summary of the ensemble is formulated based on considering $\bar{\mathbf{U}}$ as an ensemble representation. An empirical analysis on artificial and real datasets is presented to validate the proposed method.

3.3.2 Simulation Results

The experimental setup is described below, followed by the voting and aggregation results.

Parameters setup

The following default values are used in the simulations reported in this chapter. The number of runs per any setting is 25, and the distribution of the measured quantities is reported using box-plots or error bars. The value of n is set to 500 and the ensemble size is $b = 25$. The range for the generated class distributions is $p_{cl}^{i} \in [0.1, 0.5]$ (the lower limit ensures that no empty classes are generated, and the upper limit ensures that the data objects do not get assigned to one cluster).

Voting Results

First, the results of the voting loss are presented as measured by the MSE^i and Err^i defined in Sec. 3.1 for a given estimated partition $\mathbf{V}^i$ with respect to a specified reference partition $\mathbf{U}^0$. To compute Err^i for the cumulative voting scheme, $\mathbf{V}^i$ is converted to a hard partition. Note that for the voting results, no aggregation is performed. That is, the voting results only reflect the values of the voting loss functions.

For the voting results, the experiments are conducted as follows. A set of 25 partitions are generated, and one is picked at random and designated as a common reference $\mathbf{U}^0$. Then, each of the remaining partitions is optimally relabeled with respect to $\mathbf{U}^0$, using cumulative voting and

bipartite matching. Figure 3.1 illustrates the voting errors for uniform partitions with $k_i = 2$ and $k_i = 25$, and Figure 3.2 shows the voting errors for non-uniform partitions. As pointed out in Sec. 3.1, the cumulative voting scheme always leads to MSE^i and Err^i that are equal to or less than those achieved using bipartite matching. The voting results illustrate the difference between the two schemes for different types of ensembles and different error measures.

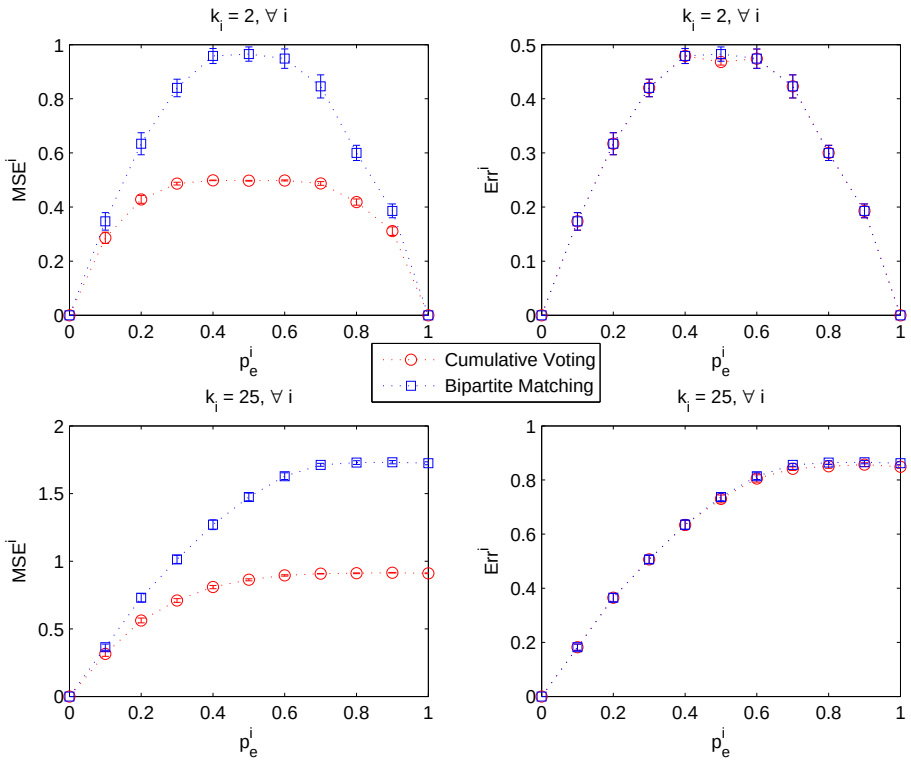

Figure 3.1: Voting loss using MSE^i and Err^i for uniform partitions with $k_i = 2$ and $k_i = 25$.

Aggregation Results

It is noted that the aggregation results illustrate the sensitivity of `Vote` to random ordering of the partitions and random selection of the initial reference $\mathbf{U}^0$; each run corresponds to one pass over the `Vote` algorithm. In real clustering problems, however, one should choose $\bar{\mathbf{U}}$ with lowest

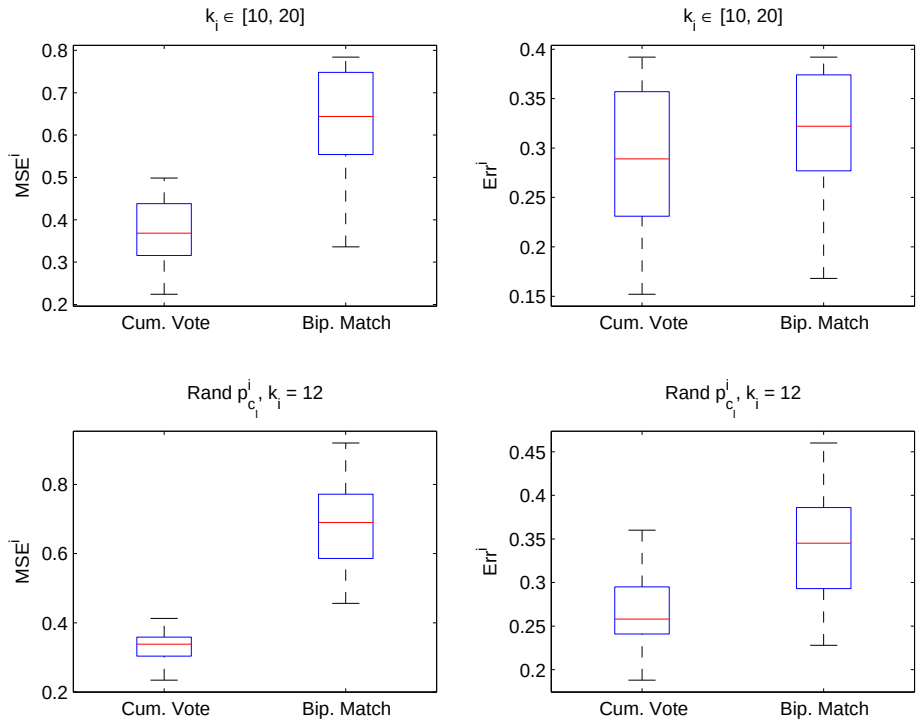

Figure 3.2: Voting loss using MSE^i and Err^i for non-uniform partitions.

MSE over all passes.

Figure 3.3 shows the $\text{MSE}(\bar{\mathbf{U}}; \mathbf{U}^1, \ldots, \mathbf{U}^b)$ defined as given by Eq. 3.11 between $\bar{\mathbf{U}}$ and the optimally re-labeled partitions $\vartheta^i(\mathbf{U}^i)$ with respect to $\bar{\mathbf{U}}$, versus p_e^i, for uniform ensembles with $k_i = 2$ and $k_i = 15$. It is noted that the MSE is not only significantly lower in the case of `cVote` compared to `bVote`, but as p_e^i increases, the MSE drops in the case of `cVote` (except for ensembles of identical partitions at $p_e^i = 0$), while it increases in the case of `bVote`.

Note that as p_e^i increases, the rate of random errors among the ensemble partitions increases, which explains the increasing MSE using `bVote`. However, for `cVote`, as p_e^i increases, the uncertainties about the assignments of the data objects increase in the estimated partition $\vartheta^i(\mathbf{U}^i)$ and the re-computed reference $\mathbf{U}^0$, at each iteration i. As the estimated cluster conditional distributions for $\mathbf{U}^0$ approach uniform distributions, they also approach uniform distributions for

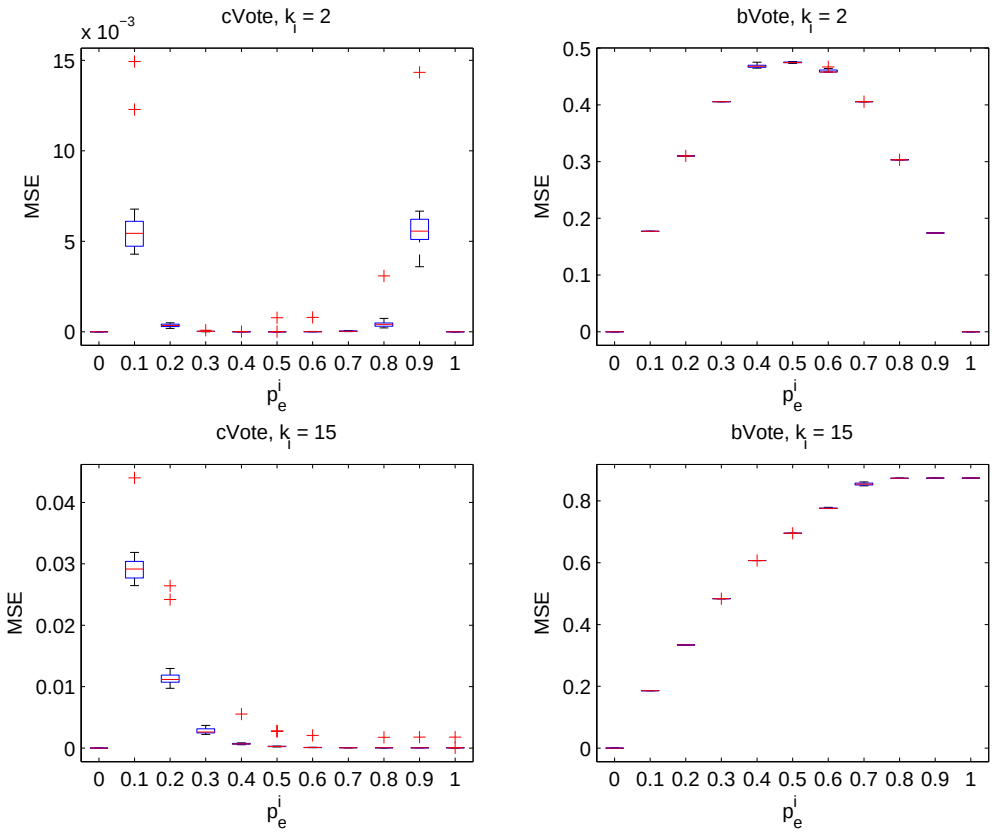

Figure 3.3: MSE versus p_e^i for uniform ensembles with $k_i = 2$ and $k_i = 15$.

$\vartheta^i(\mathbf{U}^i)$ and for $\bar{\mathbf{U}}$. In such case, the result is that MSE approaches zero. As MSE $\to 0$, the hard clustering corresponding to $\bar{\mathbf{U}}$ takes arbitrary values. Thus, it doesn't converge to $\mathbf{y}^*$, unlike the case for `bVote`, where errors tend to cancel each other.

The results presented in Fig. 3.4 and 3.5 confirm the explanation discussed above. The error rate Err* of $\bar{\mathbf{U}}$ compared to a range of possible true clusterings $\mathbf{y}^*$, versus the probability of error p_e^α of $\mathbf{y}^\alpha$ compared to $\mathbf{y}^*$, is plotted for uniform ensembles with $k_i = 2$ (Fig. 3.4) and with $k_i = 15$ (Fig 3.5) for different values of p_e^i. First, as pointed out earlier in Sec. 3.3.1 for `bVote`, $\bar{\mathbf{U}}$ is generally as good as the underlying labeling $\mathbf{y}^\alpha$ compared to $\mathbf{y}^*$, as indicated by

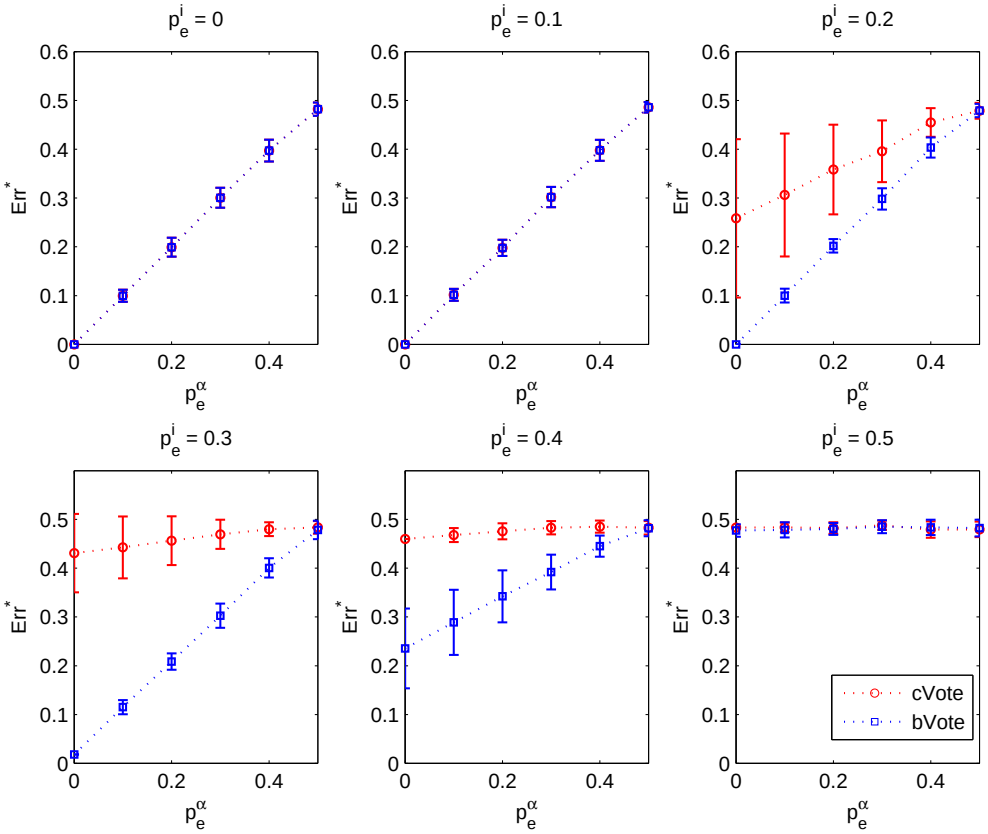

Figure 3.4: Err* versus p_e^α for uniform ensembles with $k_i = 2$, for different values for p_e^i.

the approximate equality line between Err* and p_e^α, when values of p_e^i correspond to better than random clusterings $\mathbf{y}^i$ compared to $\mathbf{y}^\alpha$. Second, it is noted that `cVote` converges to the same clustering as `bVote` at low values of p_e^i, but as p_e^i starts to increase, the increased uncertainty causes $\bar{\mathbf{U}}$ to diverge from the underlying clustering, and eventually fails in capturing a cluster structure for the data. Hence, it is concluded that for this type of ensemble, `bVote` is a more suitable relabeling scheme than `cVote`. However, it is assumed here that $k_* = k_\alpha$, but in general, the true number of clusters may be different or unknown. Such a case is further investigated in Ch. 5.

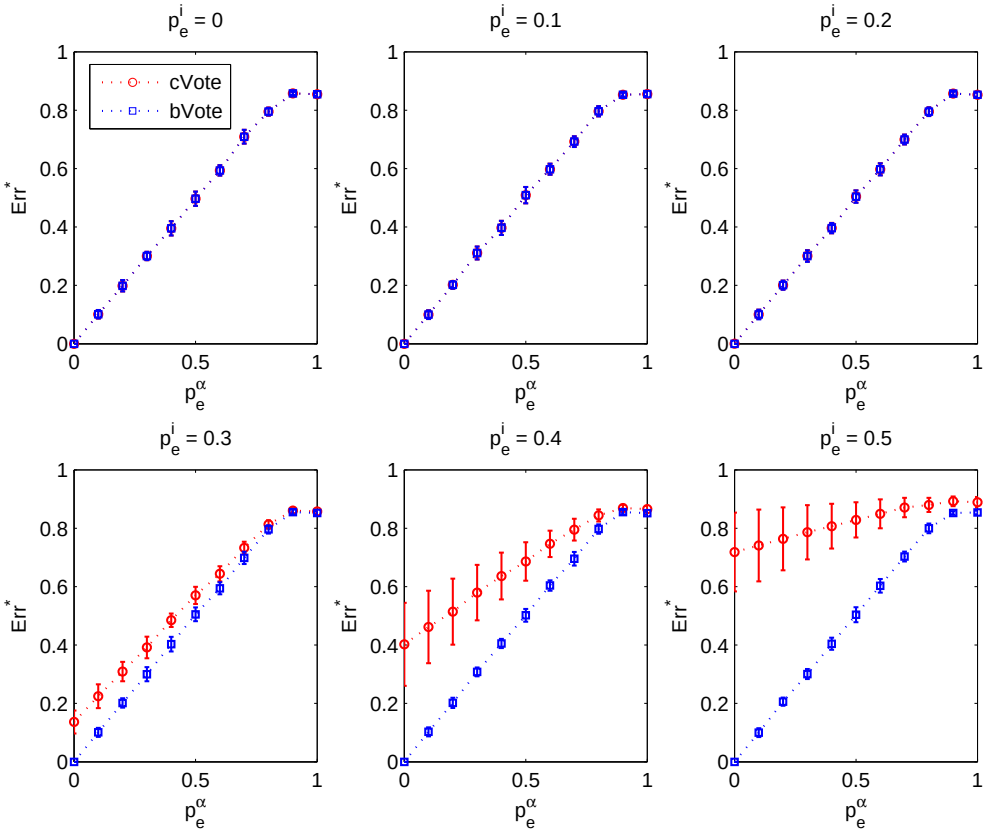

Figure 3.5: Err* versus p_e^α for uniform ensembles with $k_i = 15$ for different values for p_e^i.

Figures 3.6 and 3.7 show the MSE and corresponding Err* for non-uniform partitions. Significantly lower MSE values are obtained using `cVote` compared to `bVote`. However, the MSE values are not too low to cause `cVote` to produce arbitrary partitions compared to the initial reference partition $\mathbf{U}^0$, as indicated by Err*. In fact, lower Err* values are obtained in this case with `cVote` compared to `bVote`, which indicates that $\bar{\mathbf{U}}$ better approximates the cluster structure of $\mathbf{U}^0$, where $\mathbf{U}^0$ is considered here as a noisy replicate of $\mathbf{y}^*$. Again, note that these results are based on a particular value of $k_* = k_0$, whereas in real situations, k_* may be different or unknown. When $k^* \neq k_0$, further analysis is required to study the usefulness of $\bar{\mathbf{U}}$ based on each scheme.

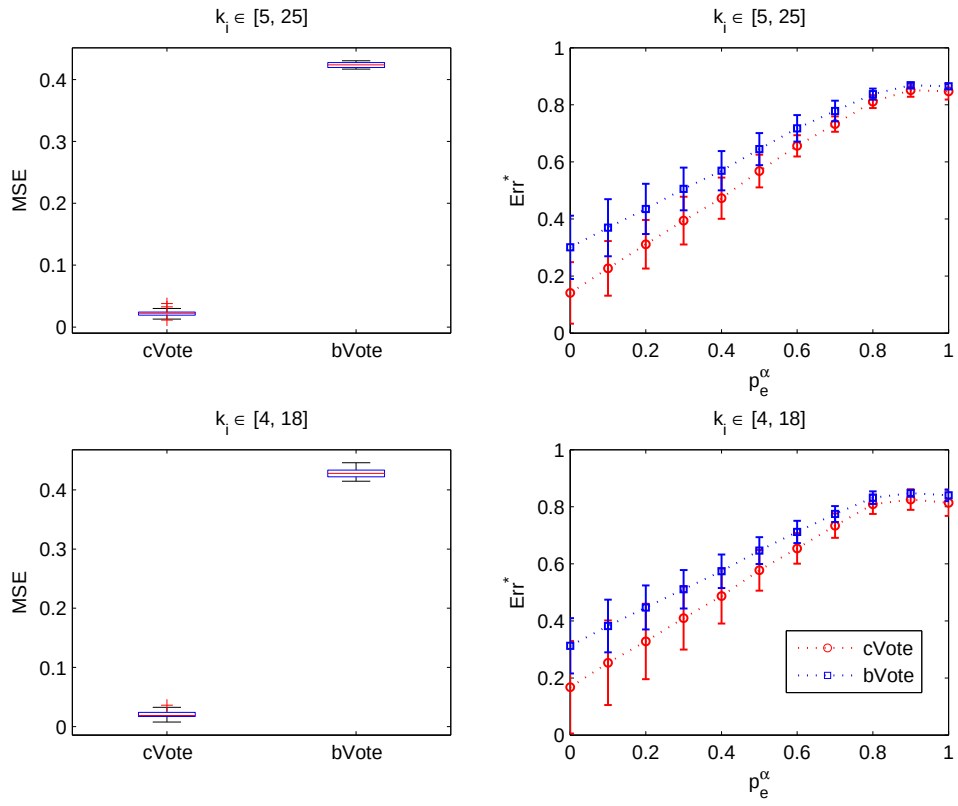

Figure 3.6: MSE and Err* for ensembles with a random number of clusters.

3.4 Discussion

It is important to note that the aggregated partition $\bar{\mathbf{U}}$ is optimized as a representation of the ensemble partitions (as given by Eq. 3.11); however, it does not necessarily reflect an accurate cluster structure for the dataset. Interpreting $\bar{\mathbf{U}}$ as a coherent and global cluster structure for the data depends on the nature of the ensemble partitions and also on the properties of the aggregation method. For instance, for ensembles of uniform partitions, $\bar{\mathbf{U}}$ is best viewed as an approximately optimal partition for the data, when the bipartite matching scheme is applied.

On the other hand, when each ensemble partition consists of a randomly selected number of

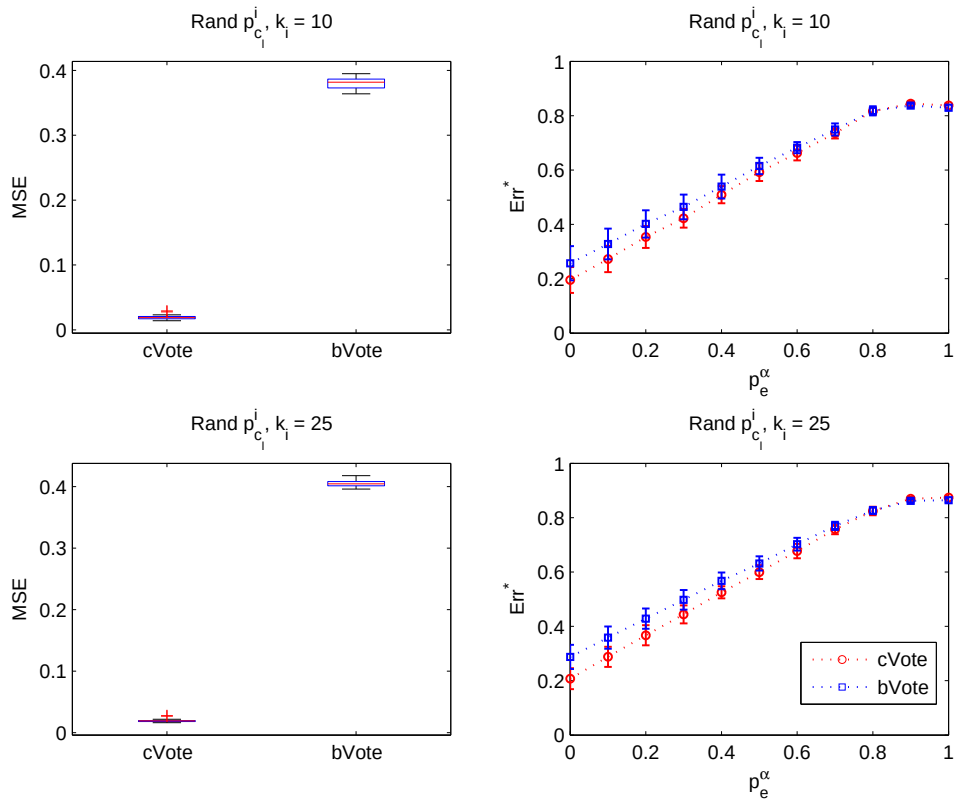

Figure 3.7: MSE and Err* for ensembles with a random cluster label distribution.

clusters, and when the true number of clusters is unknown, it is likely that $\bar{\mathbf{U}}$ does not directly represent an optimal cluster structure for $\mathcal{X}$. Instead, it may be considered as an optimized distributional representation, where the objects are characterized by the estimated conditional distribution $p(c^0|x)$. Hence, distributional clustering can be applied to find optimal partitionings of the data. In distribution clustering, objects are grouped by comparing their histograms using divergence measures [6]. The problem is generalized by Tishby et al. [8] who introduced a method referred to as the information bottleneck method.

In Ch. 5 the view of $\bar{\mathbf{U}}$ as a distributional data representation is developed further. An efficient agglomerative algorithm based on the information bottleneck formulation is proposed.

Specifically, if the desired (or meaningful) number of consensus clusters is smaller than the number of clusters in the aggregated partition, the following definition of an optimal consensus partition is proposed. It is defined as the most compressed summary of the aggregated distributional representation such that maximum amount of relevant information about the data is preserved. The proposed algorithm minimizes the average Jensen-Shannon divergence within the consensus clusters and is applied, as described in Ch. 5, for obtaining a consensus partition, for either an estimated or a pre-determined number of clusters, based on each voting scheme.

Chapter 4

On The Cumulative Voting Scheme

This chapter consists of two main sections. In Sec. 4.1, the properties of the cumulative voting scheme are investigated. In Sec. 4.2, a heuristic variant of the `cVote` algorithm is derived. The proposed algorithm seeks to maximize the mutual information associated with the empirical aggregated distribution, through adaptivity to the given ensemble partitions. It is referred to as `Ada-cVote`. Unlike `cVote`, it is characterized by the invariability of the obtained aggregated partition to the order of the ensemble partitions and the initialization of the reference partition.

4.1 Properties of Cumulative Voting

The following are notable properties of the cumulative voting schemes.

4.1.1 Unanimity Rule

It is essential for a voting-based aggregation algorithm to satisfy the unanimity rule, defined as follows. Whenever the input is an ensemble of identical partitions, modulo cluster label permutation, the output aggregated partition should also be identical to the ensemble partitions. The aggregation based on the proposed cumulative voting scheme satisfies this requirement.

Given two identical partitions $\mathbf{U}^0$ and $\mathbf{U}^i$, the coefficient matrix $\mathbf{W}^i$ computed using Eq. 3.5 constitutes a binary permutation matrix, and the estimated partition is given by $\mathbf{V}^i = \mathbf{U}^0$, representing a perfect zero-error re-labeling of $\mathbf{U}^i$ with respect to $\mathbf{U}^0$. Therefore, the averaged aggregated partition is also identical to the input partitions.

The condition of unanimity is also known as a *perfect consensus* [40]. In this case, the aggregated partition is perfectly certain, where the estimated probabilities are either 0 or 1. Note that the aggregation results in Sec. 3.3.2, for all uniform partitions with $p_e^i = 0$, illustrate this property using the `cVote` algorithm as well as the `bVote` algorithm.

4.1.2 Relation to Co-Association Matrix

Consider the co-association matrix representation of data partitions [22, 29, 31], denoted here by $\mathbf{M}$. It is a $n \times n$ matrix where each entry m_{gh} can be viewed as a vote on the co-occurrence of data objects x_g and x_h. Given an ensemble partition $\mathbf{U}^i$, a corresponding co-association matrix $\mathbf{M}^i$ is given by,

$$\mathbf{M}^i = \mathbf{U}^i \mathbf{U}^{i^T} \tag{4.1}$$

The aggregated co-association matrix of an ensemble, denoted by $\bar{\mathbf{M}}$ is given by,

$$\bar{\mathbf{M}} = \frac{1}{b} \sum_{i=1}^{b} \mathbf{M}^i \tag{4.2}$$

Consider the relabeled partition $\mathbf{V}^i = \vartheta^i(\mathbf{U}^i)$ computed using Eq. 3.2, where $\mathbf{W}^i$ is computed based on the cumulative voting scheme. The partition $\mathbf{V}^i$ can also be written as given by 4.3, where $D(\mathbf{U}^i)$ denotes a $k_i \times k_i$ diagonal matrix whose l^{th} diagonal element is equal to $1/n_l^i$.

$$\mathbf{V}^i = \left(\mathbf{U}^i D(\mathbf{U}^i) \mathbf{U}^{i^T}\right) \mathbf{U}^0 \tag{4.3}$$

The product $(\mathbf{U}^i D(\mathbf{U}^i) \mathbf{U}^{i^T})$ is a $n \times n$ doubly stochastic matrix that represents the normalized co-association matrix representation of $\mathbf{U}^i$ and is denoted here as $\widetilde{\mathbf{M}}^i$. Each entry $\widetilde{m}_{gh}^i$ is the

inverse of the cluster size to which the objects x_g and x_h belong. That is, for a pair of objects $x_g, x_h \in \mathcal{C}_l^i$ (i.e., for objects assigned cluster label c_l^i), the corresponding entry is given by $\widetilde{m}_{gh}^i = 1/n_l^i$, reflecting that objects that are members of a large cluster are less likely to co-occur together in the same cluster than objects belonging to a small cluster. According to Eq. 4.3, the entries of $\mathbf{V}^i$ represent the dot products of the row vectors of $\widetilde{\mathbf{M}}^i$ and the vectors $\mathbf{u}_{.q}^0$. However, note that it is computationally cheaper to compute $\mathbf{V}^i$ using Eq. 3.2 instead of 4.3.

Example 1.

Consider a reference partition $\mathbf{U}^0$ and an ensemble partition $\mathbf{U}^i$, as given below. Using the cumulative voting scheme, the values of $\mathbf{W}^i$ (as given by Eq. 3.5), $\mathbf{V}^i$ (as given by Eq. 3.2), and the corresponding matrix $\widetilde{\mathbf{M}}^i$ are as given below.

$$\mathbf{U}^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{U}^i = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{W}^i = \begin{bmatrix} 0.6667 & 0.3333 & 0 & 0 \\ 0 & 0.5000 & 0.5000 & 0 \\ 0 & 0 & 0.3333 & 0.6667 \end{bmatrix} \qquad \mathbf{V}^i = \begin{bmatrix} 0.6667 & 0.3333 & 0 & 0 \\ 0.6667 & 0.3333 & 0 & 0 \\ 0.6667 & 0.3333 & 0 & 0 \\ 0 & 0.5000 & 0.5000 & 0 \\ 0 & 0.5000 & 0.5000 & 0 \\ 0 & 0 & 0.3333 & 0.6667 \\ 0 & 0 & 0.3333 & 0.6667 \\ 0 & 0 & 0.3333 & 0.6667 \end{bmatrix}$$

$$\widetilde{\mathbf{M}}^i = \begin{bmatrix} 0.3333 & 0.3333 & 0.3333 & 0 & 0 & 0 & 0 & 0 \\ 0.3333 & 0.3333 & 0.3333 & 0 & 0 & 0 & 0 & 0 \\ 0.3333 & 0.3333 & 0.3333 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5000 & 0.5000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5000 & 0.5000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0 & 0 & 0 & 0.3333 & 0.3333 & 0.3333 \\ 0 & 0 & 0 & 0 & 0 & 0.3333 & 0.3333 & 0.3333 \end{bmatrix}$$

Given the relationship outlined above between the co-association matrix and the cumulative voting scheme, a variant of the cumulative voting scheme can be derived such that, in a special case, it gives an aggregated partition $\bar{\mathbf{U}}$ that is equal to the aggregated co-association matrix $\bar{\mathbf{M}}$. It turns out that this is easily obtained by simple modifications to the original cumulative voting scheme and to the aggregation algorithm applied in Ch. 3. First, instead of the normalization to 1, the rows of $\mathbf{W}^i$ should be constrained to sum to n_l^i. Thus, an un-normalized cumulative voting scheme is developed where the voting problem in 3.4 is written as follows.

$$\begin{aligned} \min_{\mathbf{W}^i} \ & \frac{1}{n}\mathrm{tr}[\left(\mathbf{U}^0 - \mathbf{U}^i\mathbf{W}^i\right)^T\left(\mathbf{U}^0 - \mathbf{U}^i\mathbf{W}^i\right)] \\ & \text{Subject to } \sum_{q=1}^{k_0} w_{lq}^i = n_l^i. \end{aligned} \tag{4.4}$$

That is, the problem corresponds to fitting a linear model by the least squares method, where one is estimating the occurrence frequencies of the objects in the representative clusters, which are described by the output variables $C^0 = \{C_1^0, \ldots, C_{k_0}^0\}$.

The solution is given by $\mathbf{W}^i = {\mathbf{U}^i}^T\mathbf{U}^0$ and an estimated co-occurrence matrix of the data objects and the reference clusters, denoted as $\grave{\mathbf{V}}^i$ is given by $\grave{\mathbf{V}}^i = \mathbf{U}^i\mathbf{W}^i$. Note that $\grave{\mathbf{V}}^i$ is equivalently expressed as given in Eq. 4.5, where the product ${\mathbf{U}^i}^T\mathbf{U}^i = \mathbf{M}^i$. However, computing $\grave{\mathbf{V}}^i$ as given above is computationally cheaper compared to Eq. 4.5, which requires $O(k_0 k_i n^2)$ instead of $O(k_0 k_i n)$.

$$\grave{\mathbf{V}}^i = \left(\mathbf{U}^i \mathbf{U}^{i^T}\right)\mathbf{U}^0 \tag{4.5}$$

As for the aggregation algorithm, in order to obtain $\bar{\mathbf{U}} = \bar{\mathbf{M}}$, the following modifications are required. First, a fixed-reference approach should be applied, instead of the iterative approach of the `Vote` algorithm where the reference partition is updated at each step. Second, the reference partitions should be the n identity matrix $\mathbf{I}_n$ (which corresponds to the n-partition of singleton clusters). Under these special conditions, the aggregated representation obtained using the un-normalized cumulative scheme is nothing but the well-known co-association matrix $\bar{\mathbf{M}}$.

Example 2.

Considering $\mathbf{U}^0$ and $\mathbf{U}^i$ as given in Example 1, $\mathbf{W}^i$ and $\grave{\mathbf{V}}^i$ are given as follows.

$$\mathbf{W}^i = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 2 \end{bmatrix} \quad \grave{\mathbf{V}}^i = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 1 & 2 \end{bmatrix}$$

The voting-based aggregation algorithm implementing the un-normalized fixed-reference cumulative voting scheme is presented in Algorithm 2, and is referred to as `URef-cVote`. The algorithm estimates the joint distribution $p(c^0, x)$, which is represented by the matrix $\grave{\mathbf{U}}$, and then uses it to compute the marginal probabilities $p(x_j)$ and the conditional distributions $p(c^0|x_j)$, where $p(c^0|x_j)$ represent the soft aggregated partition $\bar{\mathbf{U}}$.

Note that the un-normalized scheme is described here for the purpose of highlighting the relationship between the cumulative voting scheme, in general, and the co-association matrix, which is a fundamental ensemble representation. However, the analysis presented throughout

Algorithm 2 `URef-cVote`

Function $\bar{\mathbf{U}} =$ `URef-cVote`$(\mathcal{U})$

1: Randomly select a partition from $\mathcal{U}$ as a reference $\mathbf{U}^0$
2: $\grave{\mathbf{U}} = 0$, $\bar{\mathbf{U}} = 0$ {Initialize two $k_0 \times n$ matrices $\grave{\mathbf{U}}$ and $\bar{\mathbf{U}}$}.
3: **for** $i = 1$ to b **do**
4: $\quad \mathbf{W}^i = \mathbf{U}^{i^T}\mathbf{U}^0$.
5: $\quad \grave{\mathbf{V}}^i = \mathbf{U}^i\mathbf{W}^i$
6: **end for**
7: $N = \sum_{i=1}^{b}\sum_{j=1}^{n}\sum_{q=1}^{k_0} \grave{v}_{jq}^i$
8: $\grave{\mathbf{U}} = \frac{1}{N}\sum_{i=1}^{b} \grave{\mathbf{V}}^i$ {$\grave{\mathbf{U}}$ represents the empirical joint distribution $p(c^0, x)$}
9: **for** $j = 1$ to n **do**
10: $\quad P_{x_j} = \sum_{q=1}^{k_0} \grave{u}_{qj}$.
11: $\quad \bar{\mathbf{u}}_j = \grave{\mathbf{u}}_j / P_{x_j}$ {each row vector $\bar{\mathbf{u}}_j$ of $\bar{\mathbf{U}}$ represents the distribution $p(c^0|x_j)$}
12: **end for**

the thesis is focused on the normalized scheme. In [39], empirical results for the un-normalized cumulative voting scheme are presented.

4.1.3 Preserving Class Distribution

Consider the random variables $\{C^i\}_{i=1}^b$, defined over the cluster labels of each ensemble partition $\{\mathbf{U}^i\}_{i=1}^b$, with probability distribution $p(c_l^i) = n_l^i/n$ (assuming hard ensemble partitions). The optimally relabeled partition $\mathbf{V}^i$ computed with respect to a given reference partition $\mathbf{U}^0$, using the cumulative voting scheme described in Ch. 3, is a soft partition that is viewed as representing a conditional probability distribution $p^i(c^0|x)$, where the random variable C^0 is defined over the initial reference clusters $\{c_q^0\}_{q=1}^{k_0}$, and X is defined over the objects $x \in \mathcal{X}$. Suppose that the simplifying assumption that the marginal probabilities $p(x_i) = \frac{1}{n}$, $\forall j$ is made. The joint distribution $p^i(c^0, x)$, based on $p^i(c^0|x)$, can be computed using Bayes rule. Let $p(c^0)$ represent

the class distribution based on the reference partition. The class distribution $p^i(c^0)$ computed based on $\mathbf{V}^i$ is given by:

$$\begin{aligned} p^i(c_q^0) &= \sum_{j=1}^{n} p^i(c_q^0, x_j) = \sum_{j=1}^{n} p^i(c_q^0|x_j)p(x_j) \\ &= \sum_{l=1}^{k_i} p(c_q^0|c_l^i)\ \frac{n_l^i}{n} = \sum_{l=1}^{k_i} p(c_q^0|c_l^i)\ p(c_l^i) = p(c_q^0). \end{aligned} \tag{4.6}$$

That is, the class distribution of the reference clusters is preserved by the optimally relabeled partitions based on the cumulative voting scheme, $p^i(c_q^0) = p(c_q^0),\ \forall q$.

Furthermore, consider the aggregated partition $\bar{\mathbf{U}}$ computed using the cumulative voting scheme as described by the `cVote` algorithm in Algorithm 1, which represents the empirical probability distribution $\bar{p}(c^0|x)$. Since $\bar{\mathbf{U}}$ is computed by averaging $\{\mathbf{V}^i\}_{i=1}^b$, then, it follows that the probability distribution $\bar{p}(c^0)$ is equal to the class priors of the initial reference clusters, $\bar{p}(c_q^0) = p(c_q^0), \forall q$.

This means that the selection of the initial reference for the cumulative voting scheme can be used not only to specify the number of aggregated clusters $\bar{k} = k_0$, but also their probability distribution $\bar{p}(c^0)$. The usefulness of this property is further investigated in the following section. Specifically, this property is utilized to introduce a new variant of the `cVote` algorithm. The new algorithm has several advantages over the basic `cVote` algorithm. First, it maximizes the amount of information in the aggregated distributional representation of the data. Secondly, the aggregated partition becomes invariant to the ordering of the ensemble partitions and to the initial reference, thus, eliminating the random variations in the `cVote` algorithm.

It is noted that, unlike the normalized cumulative voting scheme, the un-normalized scheme does not lead to preserving the prior probabilities of the reference classes. Note that in the case of the un-normalized scheme, the distribution $p(x)$ is obtained as described by the `URef-cVote` algorithm. The objects are not considered equiprobable, instead, their probabilities are proportional to the relative sizes of the aggregated clusters.

4.2 Maximizing Information Content

In this section, the proposed aggregation algorithm is presented, followed by simulation results.

4.2.1 Rationale

The Shannon entropy $H(C)$ associated with the random variable C defined over the cluster labels of a partition $\mathbf{U}$ measures the average amount of information content associated with C and is defined as a function of its distribution $p(c)$ as follows [79], $H(C) = -\sum_{c \in C} p(c) \log p(c)$. A consequence of the property of preserving class priors for the cumulative voting scheme, the entropies associated with the class distributions for $\mathbf{V}^i$ and for $\bar{\mathbf{U}}$ are both equal to that of the initially selected reference partition $\mathbf{U}^0$.

For an ensemble partition $\mathbf{U}^i$, the value of the entropy $H(C^i)$ depends on the number of clusters k_i and the relative cluster sizes. For partitions with approximately equal-sized clusters, $H(C^i) \approx \log_2(k_i)$. The trivial partition of n singleton clusters has the maximum entropy among all possible partitions, with $H(C^i) = \log_2(n)$, while the partition with one n-sized cluster has zero entropy.

The mutual information $I(C; X)$ associated with a partition $\mathbf{U}$, measures the amount of information that the random variable C contains about X, and vice-versa, and is defined as,

$$I(C; X) = \sum_{c} \sum_{x} p(c, x) \log \frac{p(c, x)}{p(c)p(x)}, \tag{4.7}$$

and can also be written as,

$$I(C; X) \quad = \quad H(C) - H(C|X). \tag{4.8}$$

For a hard ensemble partition $\mathbf{U}^i$, it is noted that $I(C^i; X) = H(C^i)$, since the value of C^i is completely determined by the value of X (i.e., $H(C^i|X) = 0$).

Let $p(c|x)$ denote the aggregated distribution represented by $\bar{\mathbf{U}}$, i.e., $p(c|x) = \bar{p}(c^0|x)$, and let $I(C;X)$ denote the corresponding mutual information between $C = C^0$, and X. It follows that, $H(C) = H(C^0)$, and from Eq. 4.8, it is noted that $I(C;X)$ is bounded from above by $H(C)$. Hence, $I(C;X) \leq H(C^0)$.

Therefore, it follows that the initially selected reference partition for the `cVote` algorithm determines the following measures: the entropy associated with the aggregated clusters, the initial value of the mutual information $I(C^0;X)$, and the upper bound on the amount of information that random variable C contains about X, as computed based on $p(c|x)$. This result motivates the introduction of a selection criterion for the initial reference partition based on the mutual information $I(C^i;X)$, or equivalently $H(C^i)$ for hard partitions. Hence, the initial reference partition is selected as given by Eq. 4.9.

$$\mathbf{U}^0 = \arg\max_{\mathbf{U}^i \in \mathcal{U}} I(C^i;X) \equiv \arg\max_{\mathbf{U}^i \in \mathcal{U}} H(C^i) \tag{4.9}$$

Furthermore, the aggregation can be further improved if the iterative algorithm `cVote` greedily selects at each aggregation step i the ensemble partition that keeps the mutual information $I_i^0(C^0;X)$ as close as possible to $I_{i-1}^0(C^0;X)$, where $I_i^0(C^0;X)$ and $I_{i-1}^0(C^0;X)$ are associated with the reference partitions computed at step i and step $i-1$, and representing the distributions denoted here by $p_i^0(c^0|x)$ and $p_{i-1}^0(c^0|x)$, respectively. This greedy aggregation sequence limits the loss in $I(C;X)$ for the aggregated partition, unlike the random aggregation sequence of the `cVote` algorithm, which can lead to arbitrary losses in $I_i^0(C^0;X)$ and in $I(C;X)$, in turn.

Note that $p_i^0(c^0,x)$ is the average of the relabeled partitions up to the i-th iteration, as described in Algorithm 1, which is given as follows,

$$p_i^0(c^0|x) = \gamma_1\, p_{i-1}^0(c^0|x) + \gamma_2\, p^i(c^0|x), \tag{4.10}$$

where $\gamma_1 = \frac{i-1}{i}$, and $\gamma_2 = \frac{1}{i}$

The mutual information $I_i^0(C^0;X)$ is also written in terms of the *Kullback-Leibler* divergence

[79] $D(.\|.)$, (a.k.a. *relative entropy*), between the joint $p_i^0(c^0, x)$ and the product distribution $p(c^0)p(x)$ as:

$$I_i^0(C^0; X) = D(p_i^0(c^0, x) \| p(c^0)p(x)). \tag{4.11}$$

Since $p(c^0)$ and $p(x)$ remain constant for the cumulative voting scheme, the goal is to select $\mathbf{U}^i$ that leads to $p^i(c^0|x)$ being as close as possible to $p_{i-1}^0(c^0|x)$ (weighted by γ_1 and γ_2). That is, $\mathbf{U}^i$ should be selected such that the divergence between $p^i(c^0|x)$ and $p_{i-1}^0(c^0|x)$ is minimized.

A heuristic algorithm that seeks to minimize the divergence criterion is proposed. The algorithm works by choosing at each aggregation step i, the ensemble partition $\mathbf{U}^i$ that maximizes $I(C^i; X)$, or equivalently $H(C^i)$, for hard ensembles. It saves computational time as the entropies can be computed once for each partition, prior to aggregating, rather than computing, at each step i, the divergences between the current reference and all the remaining partitions, after relabeling. Furthermore, the simplified criterion represents a reasonable heuristic given the ensemble generation mechanism that is considered in this thesis. In the experimental study presented in Ch. 5, the adopted ensemble generation mechanism is as follows. The same base algorithm (the k-means), with a randomly selected number of clusters, is applied to generate the ensemble partitions. Therefore, the closer the values of $I(C^i; X)$ or $H(C^i)$, the more similar the cluster structures of $\mathbf{U}^i$, and hence, one can obtain the least amount of information loss, or equivalently, minimum divergence from the current reference distribution is obtained.

4.2.2 Adaptive Algorithm

The proposed algorithm is referred to as *adaptive cumulative voting*, (`Ada-cVote`), in reference to its "adaptivity" to a given ensemble of partitions, in such a way as to minimize the loss in the resulting mutual information. The algorithm incorporates the selection criterion for the initial reference partition as given by Eq. 4.9 as well as the greedy selection of the subsequent partitions $\mathbf{U}^i$, at each aggregation step $i = \{2, \ldots, b\}$, so as to preserve maximum amount of information.

This is simply achieved in the proposed heuristic algorithm by sorting the ensemble partitions in descending order of their entropies, $H(C^i)$, select the first partition as the initial reference and then aggregate the remaining partitions in the sorted order.

An important feature that is achieved as a by-product of the proposed adaptive algorithm is that the aggregated partition becomes invariant of the order of the input partitions and of the initial partition, unlike the `Vote` algorithm. This invariability is a generally desirable property for an aggregation algorithm and it also saves the extra computations required to enhance the `cVote` algorithm by performing multiple passes. The steps of the `Ada-cVote` algorithm are outlined in Algorithm 3.

Algorithm 3 `Ada-cVote`

Function $\bar{\mathbf{U}} =$ `I-cVote`$(\mathcal{U})$

1: Re-order $\mathcal{U}$, s.t. $\mathbf{U}^i$ are sorted in decreasing order of $I(C^i; X)$ ($\equiv H(C^i)$ for hard partitions)
2: Assign $\mathbf{U}^1$ to $\mathbf{U}^0$.
3: **for** $i = 2$ to b **do**
4: Compute $\mathbf{W}^i$ as given by Eq. 3.5.
5: $\mathbf{V}^i = \mathbf{U}^i\mathbf{W}^i$
6: $\mathbf{U}^0 = \frac{i-1}{i}\mathbf{U}^0 + \frac{1}{i}\mathbf{V}^i$
7: **end for**
8: $\bar{\mathbf{U}} = \mathbf{U}^0$.

4.2.3 Simulation Results

In this section, `cVote` and `Ada-cVote` are compared for the partition generation models described in Ch. 3. The two algorithms are compared by evaluating the obtained $I(C; X)$ and $\text{MSE}(\bar{\mathbf{U}}; \{\mathbf{U}_i\}_{i=1}^b)$ for the aggregated partitions. Furthermore, the error rates Err^* are compared in the case of uniform ensembles to investigate if the adaptive aggregation may reduce the error rate for `cVote`, which tends to perform poorly for this type of ensemble, especially as p_e^i increases (as observed in Ch. 3). Finally, the adaptive aggregation is applied to the bipartite matching

scheme and a comparison of the corresponding `bVote` and `Ada-bVote` algorithms is performed, where the latter is similar to `Ada-cVote`. The different is that bipartite matching is being applied for pairwise relabeling, in place of cumulative voting.

Figure 4.1 shows $I(C;X)$ and MSE for two instances of non uniform ensembles and one uniform ensemble. The first non-uniform ensemble has a randomly selected k_i in an arbitrary range, $k_i \in [10, 30]$, and the second has randomly generated class distributions $p^i_{c_l}$ with $k_i = 6,\ \forall i$. The uniform ensemble has $k_i = 15$ and a probability of error $p^i_e = 0.3,\ \forall i$. The simulation results consistently show higher $I(C;X)$ values with `Ada-cVote`, for the different types of ensembles. It is also noted that `Ada-cVote` leads to increased MSE values. This increase indicates lower levels of uncertainties (i.e., crisper probabilities) in the aggregated partition, which can help regulate the instabilities of the `cVote` algorithm noted in Ch 3, particularly as p^i_e increases, in the case of uniform ensembles. This hypothesis is checked by the experimental results reported in Fig. 4.2.

Figure 4.2 shows the Err^* in the case of uniform ensembles with $k_i = 15$ and where p^i_e and p^α_e are varied in $[0, 0.5]$. The results show that `Ada-cVote` indeed leads to lower error rates Err^* compared to `cVote` with respect to possible true partitions $\mathbf{U}^*$. That is, the adaptive aggregation improves the convergence properties of the aggregated partition compared to the underlying generating labeling. However, the `bVote` algorithm remains a winner, when the results are compared to the corresponding ensemble presented in Ch. 3, in Fig. 3.5.

Figure 4.3 shows $I(C;X)$ and MSE for the same ensembles considered above, but for the `bVote` and `Ada-bVote` algorithms. Unlike `Ada-cVote`, it is noted that the adaptive aggregation does not have a similar effect on the aggregated partitions for the bipartite matching scheme. Notably, in the case of uniform ensembles, the results are identical with and without the adaptive aggregation. For non-uniform ensembles, there isn't any clear increase in $I(C;X)$ and MSE as observed in the case of cumulative voting, instead, the results are comparable. Hence, the empirical evidence suggests that the adaptive aggregation is best suited for cumulative voting rather than bipartite matching.

The difference in the effect of the adaptive aggregation on each scheme can be explained by

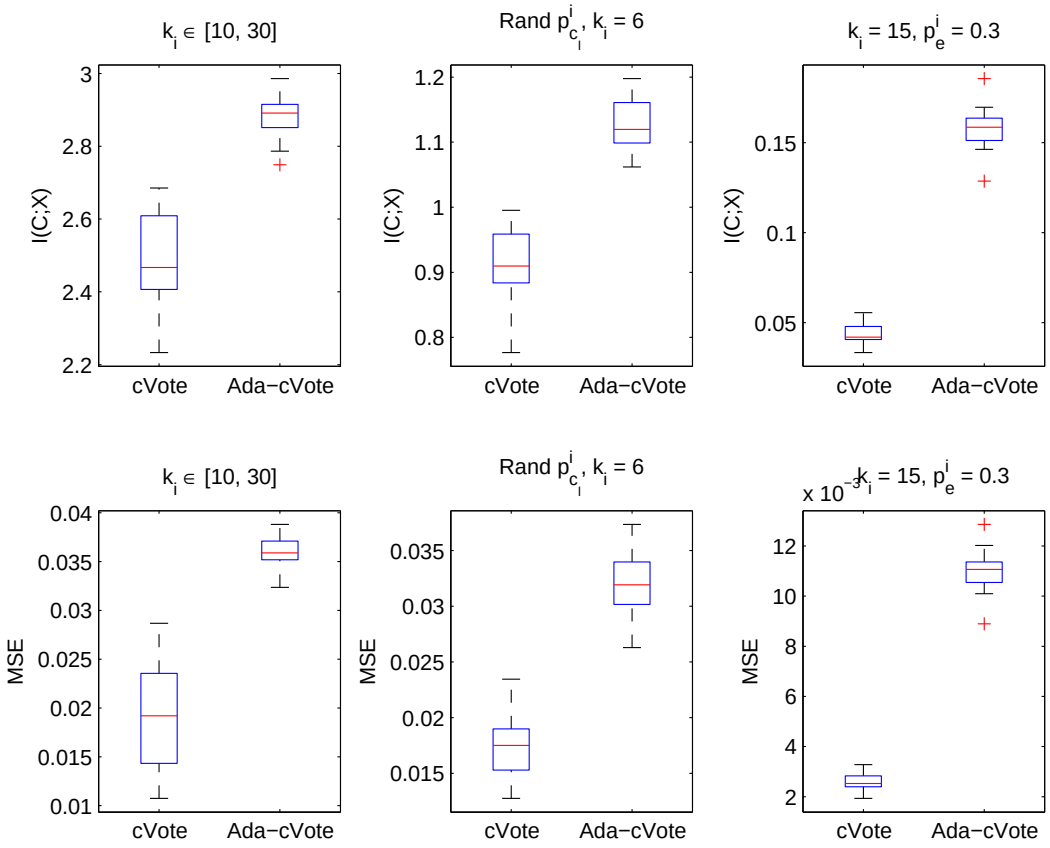

Figure 4.1: $I(C;X)$ and MSE for `cVote` and `Ada-cVote`.

the class-preserving property which is the basis for introducing the adaptive scheme and which characterizes only the cumulative voting scheme. Furthermore, it is noted that the cumulative voting alters the structure of the ensemble partition by estimating the uncertainties about the assignments of the data objects so as to best match the current representative partition, whereas, the bipartite matching simply permutes the cluster labels of the ensemble partition. Hence, the criterion for generating a particular aggregation sequence for the ensemble partitions can lead to a marked effect on the aggregated partition, only in the case of cumulative voting.

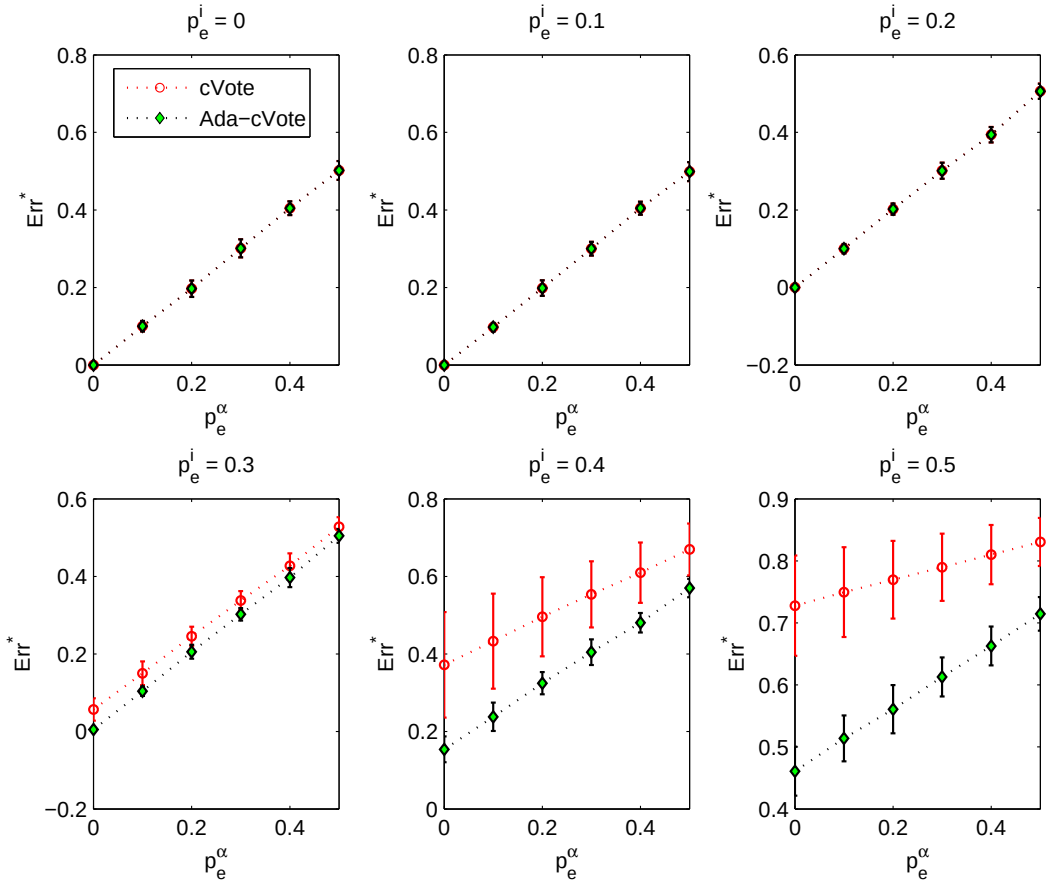

Figure 4.2: Err* for ensembles with $k_i = 15$, where $p_e^i, p_e^\alpha \in [0, 0.5]$

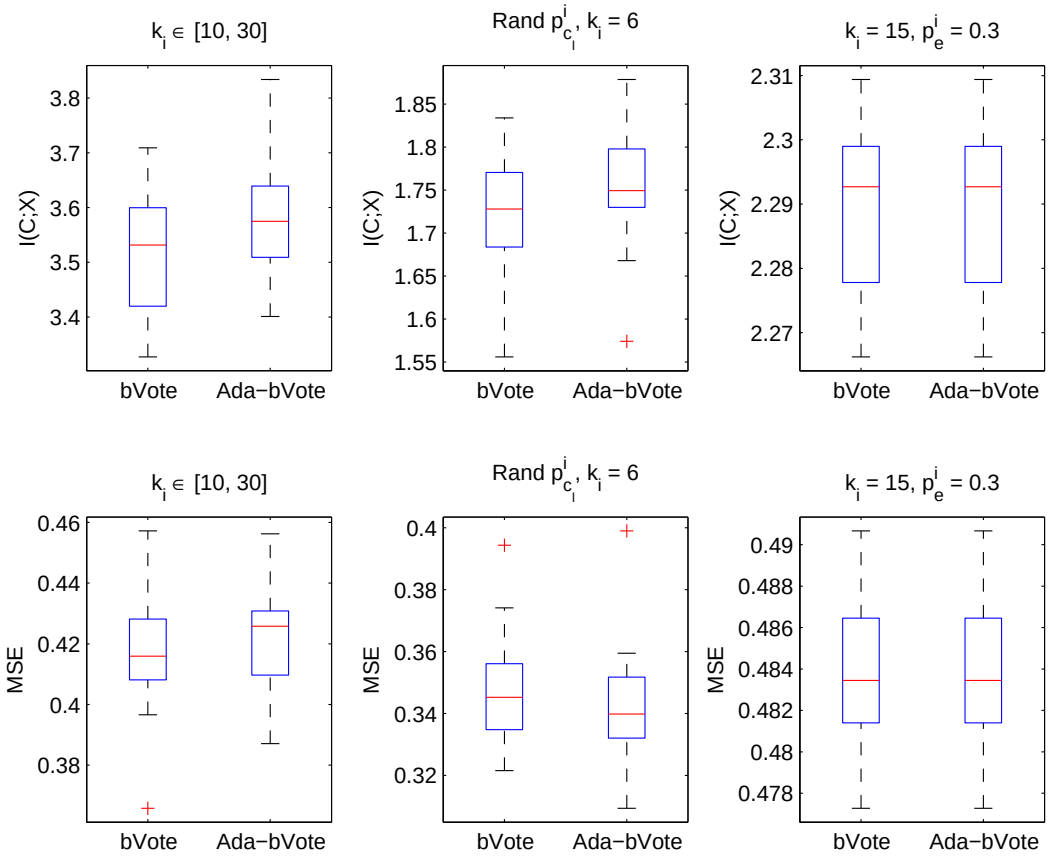

Figure 4.3: $I(C;X)$ and MSE for `bVote` and `Ada-bVote`.

Chapter 5

Compression of Aggregated Representation

The goal of this chapter is to further develop the idea of considering the aggregated partition as an optimized distributional representation of the ensemble and apply a principled information theoretic approach to extracting a coherent and global cluster structure for the data. In Sec. 5.1, the theoretical basis is presented. In Sec. 5.2, a computationally efficient approximation algorithm is proposed, which is based on the introduced theoretical basis for finding an optimally compressed consensus partition. Furthermore, an approach to estimating an optimal number of clusters is presented. In Sec. 5.3, an empirical study is presented for validating the proposed consensus method. Finally, a summary is presented in Sec. 5.4.

5.1 Theoretical Basis

The theoretical considerations underlying the proposed consensus extraction algorithm are discussed in this section.

5.1.1 The Information-Bottleneck Method

Consider the distributional representation of the data as given by the aggregated partition $\bar{\mathbf{U}}$, which represents the empirical probability distribution $p(c|x)$, where C is defined here as a random variable over a set of clusters $c_q \in C$, and X is the random variable defined over the space of data objects $x_j \in X$. Assuming that $p(x) = \frac{1}{n}$, $\forall x$, the joint distribution $p(c, x)$ can be computed using the Bayes rule. The number of clusters $\bar{k}$ in $\bar{\mathbf{U}}$ will be simply denoted here as k.

Clustering of distributional data, referred to as distributional clustering, was investigated by Pereira et al. [80], for clustering of words according to their distribution in particular syntactic contexts, where words are represented by the relative frequency distributions of contexts in which they appear. The relative entropy [79] between the distributions, which is also known as the Kullback-Leibler divergence, is used as the similarity measure for clustering. The problem is generalized by Tishby et al. [8] who introduced a method referred to as the Information Bottleneck (IB) method.

The IB method of Tishby et. al. [8] defines a principle that deals with the extraction of an efficient representation of relevant information. It is applicable in a variety of learning problems and provides an approach for quantifying the notion of *relevant* information. Given the joint statistics of two random variables, say X and C, one searches for a relevant quantization to compress X as much as possible while capturing as much information as possible about C, which is designated as the relevance variable. Since compression leads to loss of information compared to the original data, a trade-off is sought between compressing the representation and preserving relevant information. This is viewed as passing the mutual information between the two random variables through a "bottleneck" formed by compact representations of $\tilde{X}$ [8].

That is, given $p(x, c)$ as input, one looks for a compressed representation of X, denoted $\tilde{X}$, that maximizes the amount of information about C in $\tilde{X}$, while maximizing the compression of X. The amount of information about C in $\tilde{X}$ is measured by the mutual information $I(\tilde{X}; C)$.

Slonim and Tishby [81] describe the problem as that of finding a mapping $p(\tilde{x}|x)$ that mini-

mizes the lossy coding length of X via $\tilde{X}$, $I(X;\tilde{X})$, under a constraint on the mutual information to the relevance variable $I(\tilde{X};C)$. The problem is formulated as a minimization of the following Lagrangian [8],

$$\mathcal{L}[p(\tilde{x}|x)] = I(\tilde{X};X) - \beta I(\tilde{X};C), \tag{5.1}$$

where β is the Lagrangian multiplier. At $\beta = 0$, one gets the most compression, mapping all $x \in X$ to a single element $\tilde{x}$, and as $\beta \to \infty$, the most detailed representation is obtained [8].

Dhillon et al. [82] used an information-theoretic formulation similar to the IB method and proposed a criterion based on the generalized Jensen-Shannon divergence [83] for word clustering. They developed a divisive algorithm that minimizes the within-cluster Jensen-Shannon (JS) divergence while simultaneously maximizing the between cluster JS divergence.

5.1.2 Mutual Information and Jensen-Shannon Divergence

Slonim and Tishby present in [81] a decision-theoretic interpretation of the information bottleneck method. Consider a decision problem with k classes $C = \{c_1, \ldots, c_k\}$, with prior probabilities $\{p(c_i)\}$, and class conditional distributions $p(x|c_i)$ (where $p(x|c_i)$ is written here as $p_{c_i}(x)$). The generalized JS divergence [83] of k class distributions is defined as given by Eq. 5.2.

$$\mathrm{JS}_{p_{c_i}}\big(p_{c_1}(x), \ldots, p_{c_k}(x)\big) = H\big(\sum_{i=1}^{k} p(c_i) p_{c_i}(x)\big) - \sum_{i=1}^{k} p(c_i) H(p_{c_i}(x)) \tag{5.2}$$

where $H(p(x))$ is the Shannon entropy, also written as $H(X)$. The JS divergence between two distributions is symmetric, bounded, non-negative and equal to zero when $p_{c_i}(x) = p_{c_j}(x),\ \forall c_i, c_j$.

Slonim et al. [81] observe that, in a decision theoretic problem, the JS divergence can be written as,

$$\mathrm{JS}_{p_{c_i}}\big(p_{c_1}(x), \ldots, p_{c_k}(x)\big) = H(X) - H(X|C) = I(X;C) \tag{5.3}$$

That is, the JS divergence of the conditional distributions is equal to the mutual information between the object space X and the class space C. Therefore, constraining the JS divergence is equivalent to constraining the mutual information.

Lin [83] shows that the Bayes probability of error given by 5.4, is bounded from above and below by the JS divergence of the class conditional distributions $p(x|c_i)$, as given by Eq. 5.5.

$$\Pr_{\text{Bayes}}(err) = \sum_{x \in X} p(x)(1 - \max_i p(c_i|x)) \tag{5.4}$$

$$\frac{1}{4(k-1)}(H(C) - \text{JS}_{p_{c_i}}(p_{c_i}(x))^2 \leq \Pr_{\text{Bayes}}(err) \leq \frac{1}{2}(H(C) - \text{JS}_{p_{c_i}}(p_{c_i}(x)). \tag{5.5}$$

That is, bounds on the Bayesian probability of error are obtained by constraining the JS divergence [81]. This motivates an approach to the information bottleneck problem proposed in [81] that is based on a greedy bottom-up merging, referred to as Agglomerative Information Bottleneck algorithm (abbreviated here by AIB). Given as input $p(x, c)$, the algorithm computes a hierarchy of m-partitions of X, denoted here as $\tilde{X}_m$, for $1 \leq m \leq n$. At each step, a greedy merge of components of the current partition that minimizes the loss of mutual information is performed, which is achieved by merging the components with the minimum JS divergence between the corresponding conditional distributions ($p(c|\tilde{x})$). The algorithm further provides a measure of efficiency $I(\tilde{X}_m; X)$ whereby the quality of each $\tilde{X}_m$ partition is defined as the fraction of the mutual information between C and X that $\tilde{X}_m$ captures. The result is a curve of $I(\tilde{X}_m, X)/I(C, X)$ versus m. The loss in mutual information $\delta(m)$, at each merging step is given as $\delta(m) = \frac{I(\tilde{X}_m;X) - I(\tilde{X}_{m-1};X)}{I(C;X)}$. A drop in $\delta(m)$ indicates that a meaningful value for m was reached and that further merging results in significant loss of mutual information. The algorithm has a complexity of $O(n^2)$.

5.2 Efficient Agglomerative Algorithm

In this section, the proposed algorithm for the extraction of a consensus partition based on the aggregated distributional representation of the ensemble is described.

5.2.1 Formulation

Consider the problem of extracting an optimal partition $\hat{\mathbf{U}}$ with $\hat{k}$ clusters, based on the estimated probability distribution $p(c|x)$, as given by the aggregated partition $\bar{\mathbf{U}}$. This problem is formulated as the converse of the IB problem defined in Sec. 5.1.1 and addressed in Sec. 5.1.2. Specifically, one seeks a compressed representation of the random variable C (instead of X), denoted $\tilde{C}$, that maximizes the amount of information about X in $\tilde{C}$, $I(\tilde{C};X)$, while maximizing the compression of C. This leads to limiting the search for $\hat{\mathbf{U}}$ to values of $\hat{k}$ such that $1 \leq \hat{k} \leq k$, rather than $1 \leq \hat{k} \leq n$, where $k \ll n$. The proposed formulation allows us to develop an agglomerative JS divergence based algorithm that is computationally more efficient than the AIB algorithm. The algorithm draws on the theoretical considerations presented earlier, while having a linear computational complexity in n.

Note that for a $O(n^2)$ computational cost, one can use co-association-based consensus algorithms, which provide a simpler and competitive approach compared to voting-based consensus methods. So, one of the main advantages of the proposed method is the linear complexity in n, which makes it attractive for clustering problems with a large number of data objects. The limitation is that the possible number of clusters for the sought partition $\hat{\mathbf{U}}$ is assumed to be not larger than the number of clusters of the aggregated partition.

Consider the conditional distributions $p_{c_i}(x) = p(x|c_i)$, and prior probabilities $p(c_i)$. Let a pair of distributions be denoted as $p_{c_l}(x)$, and $p_{c_q}(x)$, $\forall l, q \in \{1, \ldots, k\}$, $l \neq q$, with priors $p(c_l)$ and $p(c_q)$, respectively. Let the weights β_l and β_q be given by $\beta_l = \frac{p(c_l)}{p(c_l)+p(c_q)}$, and $\beta_q = \frac{p(c_q)}{p(c_l)+p(c_q)}$. The JS divergence between two distributions $p_{c_l}(x)$ and $p_{c_q}(x)$ is given by [83],

$$\mathrm{JS}_\beta(p_{c_l}(x), p_{c_q}(x)) = H(\beta_l p_{c_l}(x) + \beta_q p_{c_q}(x)) - \beta_l H(p_{c_l}(x)) - \beta_q H(p_{c_q}(x)). \tag{5.6}$$

The proposed algorithm follows an agglomerative greedy approach, where the distributions $\{p(x|c)\}$ are merged incrementally at each step. Note that unlike the AIB algorithm, which merges the distributions $\{p(c|x)\}$, the cost of computing the JS divergences for $\{p(x|c)\}$ is $O(n)$, using the proposed algorithm. So, to further cut down the computation time, JS_β is computed between each pair of the k distributions, producing $k(k-1)/2$ divergences. The computed pairwise divergences are then used as input to a hierarchical algorithm with a suitable objective. The group average (average link) is uses, which, at each merging step, minimizes the average pairwise divergences between the members' distributions of the merged clusters.

The proposed algorithm starts with a k-partition where each distribution $p(x|c_i) \in \{p(x|c_i)\}_{i=1}^{k}$ is assigned to a singleton cluster. It produces a hierarchy of $\tilde{k}$-partitions, for $1 \leq \tilde{k} \leq k$, each corresponding to the compressed representation of X and a $\tilde{k}$-class variable denoted here by $\tilde{C}_{\tilde{k}}$. At a given level $\tilde{k}$ of the hierarchy, let S_g and S_h denote any two clusters of distributions, with cardinalities $|S_g|$ and $|S_h|$, respectively. That is, a cluster S_g is assigned a number $S_g \geq 1$ of distributions $\{p(x|c_l^g)\}_{l=1}^{|S_g|}$. The cost function minimized by the average link algorithm is written as given by Eq. 5.7.

$$\frac{1}{|S_g||S_h|} \sum_{p(x|c_l^g)\in S_g} \sum_{p(x|c_r^h)\in S_h} \mathrm{JS}_\beta(p(x|c_l^g), p(x|c_r^h)). \tag{5.7}$$

By minimizing the average JS divergence within the cluster, at each merging step, the algorithm approximately minimizes loss of mutual information $I(\tilde{C}; X)$ as shown in [81]. Furthermore, minimization of the classification error is achieved as a result of the bounds established in [83], as outlined above.

Like the AIB algorithm, a measure of efficiency of the obtained representations at each level of the hierarchy can be defined to estimate an optimal number of clusters. The defined measure is described below.

5.2.2 Optimal Partition Estimation

Given the hierarchical partitioning obtained by the agglomerative algorithm, one may be interested in extracting a partition $\hat{\mathbf{U}}$ with a pre-specified number of clusters, or also in estimating an optimal number of clusters, where $2 \leq \hat{k} \leq k$. First, the computation of the partition $\hat{\mathbf{U}}$, given that $\hat{k}$ has been determined (either estimated or pre-specified) is outlined. Then, the proposed approach for estimating an optimal number of clusters is described.

Given a value for $\hat{k}$, the dendrogram is cut at $\hat{k}$, and the $\hat{k}$-partition is obtained as follows. Let $\{S_g\}_{g=1}^{\hat{k}}$ denote the clusters of merged distributions at level $\hat{k}$ of the hierarchy. The priors for the $\hat{k}$ classes, denoted by $\{p(\tilde{c}_i)\}_{i=1}^{\hat{k}}$ are computed as follows:

$$\hat{p}(\tilde{c}_i) = \sum_{1 \leq l \leq |S_g|} p(c_l^g). \tag{5.8}$$

The joint distributions $\{p((x, \tilde{c}_i)\}_{i=1}^{\hat{k}}$ are computed as given by Eq. 5.9

$$\hat{p}(x, \tilde{c}_i) = \sum_{1 \leq l \leq |S_g|} p(x|c_l^g)p(c_l^g) \tag{5.9}$$

Using 5.9, $\hat{\mathbf{U}}$ is obtained as the estimated conditional distribution $p(\tilde{c}|x)$, and a hard labeling $\hat{y}$ can be obtained by assigning each $x \in X$ to its most likely cluster $\tilde{c} \in \tilde{C}$.

As for estimating an optimal value for $\hat{k}$, the idea of a $\tilde{k}$-cluster *lifetime* described in [25] is applied on the merging JS divergence thresholds. Specifically, the optimal $\hat{k}$ is defined as the number of clusters with the longest lifetime, where the lifetime of each $\tilde{k}$ is defined as the range of distance threshold values that lead to a $\tilde{k}$-partition solution. It is computed as the difference between the minimum and maximum distances that lead to merging the input patterns into $\tilde{k}$ clusters. In other words, the lifetime of $\tilde{k}$ is the difference between the merging distance leading to a $\tilde{k}$-partition and that leading to a $(\tilde{k}-1)$-partition in the obtained dendrogram.

The proposed agglomerative algorithm, referred to as `JS-ALink`, is outlined in Algorithm 4. It takes as input the estimated distribution $p(c, x)$ based on the aggregated partition $\bar{\mathbf{U}}$ and

computes a hierarchy of $\tilde{k}$-partitions as described above. There is an option to specify a value for $\hat{k}$, where $\hat{k} < k$. If no value of $\hat{k}$ is given as input, the algorithm estimates the optimal $\hat{k}$ corresponding to the most efficient partition of the data which preserves the maximum amount of relevant information, as discussed in the theoretical basis presented earlier in this chapter.

Algorithm 4 The `JS-ALink` Algorithm

Function $\hat{\mathbf{U}}$ = `JS-ALink`$(p(c,x), [\hat{k}])$

1: Compute JS_β divergence between each pair of the distributions $\{p(x|c_i)\}_{i=1}^{k}$ using Eq. 5.6
2: Apply the hierarchical average link algorithm on the list of $\frac{k(k-1)}{2}$ JS_β divergences.
3: **if** $\hat{k}$ is Not specified **then**
4: Find $\hat{k}$ with longest lifetime
5: **end if**
6: Estimate $\hat{\mathbf{U}}$ Using Eqs. 5.8 and 5.9

5.3 Empirical Study

To validate the proposed cumulative voting algorithm `Ada-cVote` in conjunction with the agglomerative `JS-Alink` algorithm, experimental results on artificial and real datasets are presented. Furthermore, the `JS-ALink` algorithm is applied in conjunction with the `bVote` algorithm in order to compare the two voting schemes. In both cases, the algorithms are evaluated for extracting a partition with a pre-determined number of clusters and for estimating an optimal number of clusters and a corresponding partition.

Moreover, the performance of the voting-based algorithms is compared with several recent consensus algorithms using external and internal measures. Sec. 5.3.1 outlines the consensus algorithms used in the comparative evaluation. Sec. 5.3.2 presents a description of the datasets. The ensemble generation mechanism is described in Sec. 5.3.3. In Sec. 5.3.4, the adopted performance evaluation measures are defined. The experimental results are presented in Sec. 5.3.5 for artificial datasets and in Sec. 5.3.6 for real datasets. Finally, in Sec. 5.4, a summary of

the empirical study is presented.

5.3.1 Consensus Algorithms

The `Ada-cVote` combined with the `JS-Alink` algorithm is abbreviated by ACV (Adaptive Cumulative Voting), in the case where the optimal number of clusters $\hat{k}$ is being estimated. On the other hand, when the number of clusters is pre-determined, the combined algorithms are denoted as ACV-k. Similarly, when applying the `bVote` with the `JS-Alink` algorithm, the consensus algorithm is abbreviated by BV (Binary/Bipartite Voting), in the case where $\hat{k}$ is being estimated. When $\hat{k}$ is pre-determined, it is abbreviated by BV-k.

The other consensus algorithms applied in the comparative evaluation are described below. All of the algorithms have been implemented using MATLAB.

- The evidence accumulation consensus (EAC) algorithms [25], where each of the hierarchical single link and average link algorithms are applied on the co-association matrix. The corresponding algorithms are referred to as EAC-S and EAC-A for the single and average link, respectively. The EAC algorithms were implemented as follows. The co-association matrix is computed and a distance function ($1-$co-association ratio) is calculated and used as input for the hierarchical algorithms.

- The graph-based algorithms: CSPA, HGPA, and MCLA [29]. The implementation provided at the authors' website[1] was used.

- The quadratic mutual information algorithm, QMI [30,38]. The algorithm was implemented as specified by the authors. A standardization is applied to transform the cluster labels into quantitative features by replacing the i-th partition by k_i binary features. Then, each binary feature is standardized to a zero mean. The k-means algorithm is applied on the

[1] http://www.strehl.com/

transformed data to find the consensus clustering (10 runs for the k-means algorithm are performed and clustering with minimum mean squared error is selected).

5.3.2 Data Sets

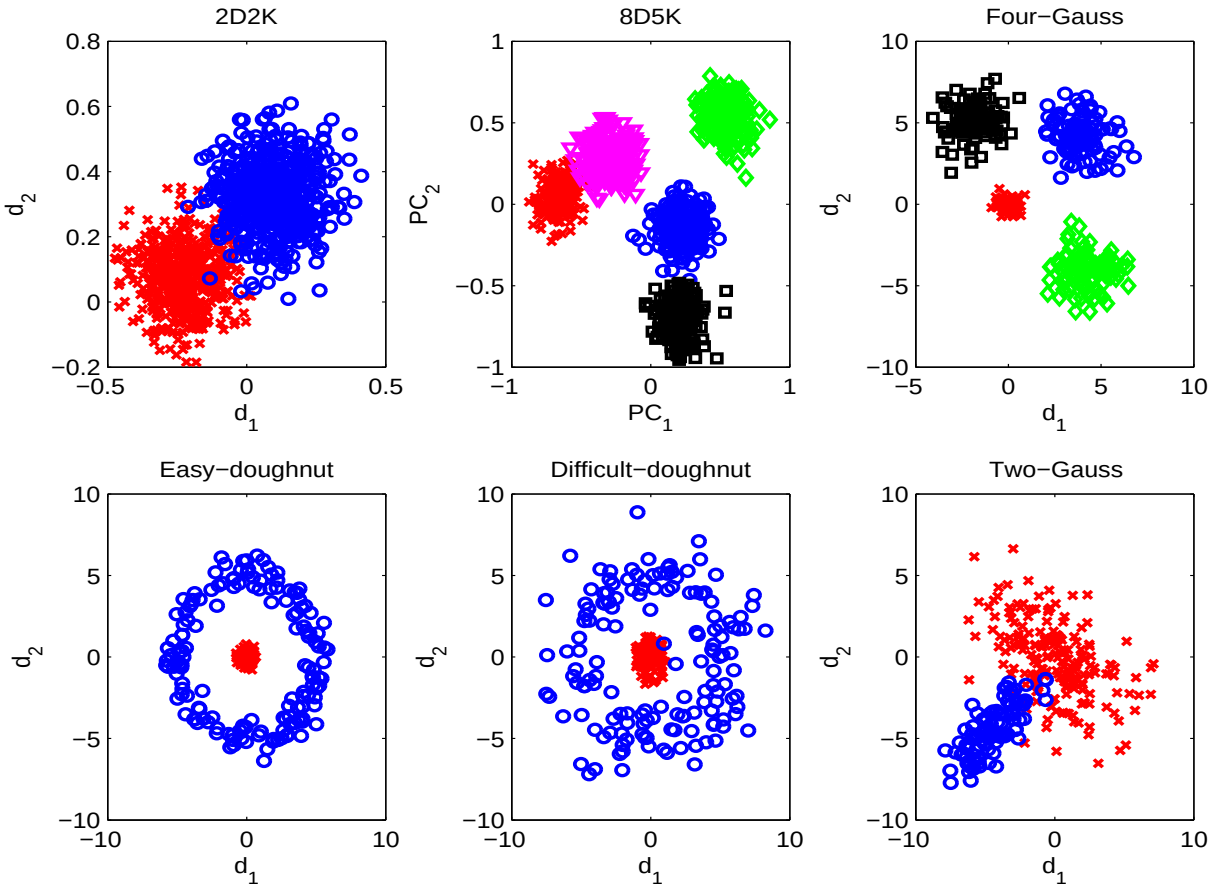

Figure 5.1: Artificial datasets. Each cluster indicated by a distinct symbol.

Table 5.1 summarizes the characteristics of the datasets used in the experiments, along with the accuracy as measured by the adjusted Rand Index [13] for the k-means algorithm (or spherical k-means for text data) compared to the true clustering, where k is set to the true number of

clusters. The mean and standard deviation (std) over 25 runs are shown.

Two-dimensional plots of the artificial datasets are shown in Fig. 5.1, where the 8D5k dataset is projected onto the first two principal components, whereas the four Gauss, easy doughnut, and difficult doughnut datasets are plotted against the first two dimensions. Below are further description of the datasets.

Table 5.1: Characteristics of the datasets and ARI values for the k-means (mean $\pm$ std).

Dataset	n	d	k	**Class distribution**	**k-means ARI**
Artificial Datasets					
2D2K	1000	2	2	50% each	0.92 ± 0.00
8D5K	1000	8	5	20% each	0.86 ± 0.16
Four Gauss	400	12	4	25% each	0.81 ± 0.21
Easy doughnut	300	12	2	50% each	0.15 ± 0.02
Difficult doughnut	300	12	2	50% each	0.15 ± 0.01
Two Gauss	300	2	2	33%-67%	0.81 ± 0.00
Real Datasets					
Breast cancer	683	9	2	65%-35%	0.84 ± 0.00
Optical digits	500	64	10	$\sim$ 10% each	0.60 ± 0.05
Yahoo!	2340	1458	6	$\sim$ 6-59-21-5-6-3%	0.42 ± 0.08

For the artificial data, five datasets that were generated and used in previous related work are used, and one additional dataset is designed. They are described as follows.

- The 2D2K and 8D5K datasets[2] were generated in [29]. The 2D2K consists two 2-dimensional Gaussian clusters with 500 points each. The clusters have different means and equal variance and they slightly overlap. The 8D5K consists of 1000 points generated from 5 8-d Gaussian distributions (200 points each). Clusters have the same variance, and means were drawn from a uniform distribution within the unit hypercube [29].

- The Four Gauss, easy doughnut, and difficult doughnut datasets[3] were generated in [36] .

[2]Datasets available at http://www.strehl.com/

[3]MATLAB functions for generating the datasets are available from http://www.informatics.bangor.ac.uk/~kuncheva/

They are 12-d, where the first 2 dimensions are meaningful, while the remaining 10 consist of uniformly generated random noise.

- The Two Gauss dataset is a mixture of two Gaussian clusters with unbalanced sizes (100 and 200 points), different means and covariance matrices, and a slight overlap.

For the real data, the following publicly available datasets are used:

- The Wisconsin breast cancer dataset [84] is available from the UCI machine learning repository. It consists of 683 patterns (after the removal of patterns with missing values) in 9 integer-valued dimensions with values in $[1, 10]$. The dataset contains 2 classes with 444 designated as benign and 239 as malignant samples.

- The optical recognition of handwritten digits (optical digits) dataset is available from the UCI machine learning repository. A random sample of size 500 patterns is generated from the training set. Patterns are described by 64 integer-valued features in $[0, 16]$. The data has 10 classes representing each digit, which are approximately equal sized.

- The Yahoo! dataset[4]. It consists of 2340 documents parsed from Yahoo! news web-pages. It has 6 classes: business, entertainment, health, politics, sports, and technology. The dataset has a high dimensionality, and the problem is compounded by highly unbalanced class sizes. There is also an alternative classification for this dataset which consists of 20 classes, whereby the Entertainment class is subdivided as follows; no sub-category, art, cable, culture, film, industry, media, multimedia, music, online, people, review, stage, television, variety. That is, 15 out of 20 classes are subclasses of the superclass "Entertainment". The subgroups are as small as 9 documents per class. In addition, there are subgroups designated as "no subcategory" and "variety" which doesn't indicate classes with a consistent content. Other subgroups have labels that indicate some similar content. Hence, the 6-class categorization is considered the true clustering.

[4]Data available at http://ftp.cs.umn.edu/dept/users/boley/PDDPdata/doc-K/

5.3.3 Ensemble Generation Technique

The ensemble generation techniques proposed in [25, 36] are applied here. The ensembles consist of partitions with overproduced clusters (i.e., k_i is larger than the desired or suspected number of clusters). Furthermore, ensembles where the number of overproduced clusters is randomly selected for each partition [25, 36] are also generated. In this case, k_i is generated randomly in a range $[k_{\min}, k_{\max}]$ that is specified as input. This strategy induces higher variability among the ensemble partitions. In general, the range $[k_{\min}, k_{\max}]$ can be varied in a search for stable consensus partitions for each dataset. In the experiments, $k_{\min}$ is usually selected as a multiple of the desired number of clusters, and $k_{\max}$ is set to any relatively larger value.

The k-means algorithm with Euclidean distance (or with the cosine measure in the case of text data) is applied as the base clustering algorithm. By default, $b = 25$, and the number of runs per any setting is 25. For the `bVote` algorithm, 10 passes over the algorithm are performed and the aggregated partition $\bar{\mathbf{U}}$ with the minimum MSE value is used subsequently in conjunction with the `JS-ALink` algorithm. For datasets with sizes $n \leq 1000$, all consensus algorithms outlined earlier are applied. However, for larger datasets, co-association based algorithms with quadratic complexity in n are excluded.

Note that for ensembles with k_i constant for all i and equal to the desired number of clusters in the consensus partition, and where the partitions are generally uniform (i.e., they are generated using the same base clustering algorithm), the simulation-based analysis demonstrated that the bipartite matching scheme is more suitable than the cumulative voting scheme. Therefore, the focus in the empirical analysis presented in this chapter is on the alternative ensemble generation mechanisms described above.

5.3.4 Performance Evaluation

To evaluate the quality of the consensus partition extracted by the different consensus algorithms, one external and one internal evaluation measure are used, both of which are widely applied in

the data clustering literature. For the external measure, the adjusted Rand index (ARI) [13] is used as a measure of agreement between the extracted partition and the true clustering for the dataset, which is available externally. For the internal measure, the Average Normalized Mutual Information (ANMI) defined in [29] between the hard consensus partition and members of the ensemble $\{\mathbf{U}^i\}_{i=1}^b$ is used. The normalized mutual information NMI is a pairwise measure of the statistical information shared between two different clusterings represented as categorical random variables.

The ARI measure is computed as follows. Let the true partition be denoted by $\mathbf{U}^*$, and let $\hat{\mathbf{U}}$ denote the extracted consensus partition (after conversion to a hard partition, in the case of voting-based algorithms). Let n_{lq} denote the number of objects that are in both the l-th cluster of $\mathbf{U}^*$, and the q-th cluster of $\hat{\mathbf{U}}$. Let $n_{l.}$ and $n_{.q}$ denote the number of objects in the l-th cluster of $\mathbf{U}^*$ and the q-th cluster of $\hat{\mathbf{U}}$, respectively. The general form of the index is given by $\frac{\textit{index - expected index}}{\textit{maximum index - expected index}}$. The expected value of ARI is zero and its maximum value is 1. Hence, there is a wide range of values that the ARI index can take compared to measures taking values between 0 and 1, thus increasing the sensitivity of the index [13]. The ARI takes the value 0 when the index equals its expected value. It is defined as given below in Eq. 5.10.

$$\text{ARI} = \frac{\sum_{l,q}\binom{n_{lq}}{2} - \left[\sum_l \binom{n_{l.}}{2}\sum_q \binom{n_{.q}}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_l \binom{n_{l.}}{2}\sum_q \binom{n_{.q}}{2}\right] - \left[\sum_l \binom{n_{l.}}{2}\sum_q \binom{n_{.q}}{2}\right]/\binom{n}{2}}. \tag{5.10}$$

The ANMI measure is defined as follows. Let n_{lq}^i denote the number of objects that are in both the l-th cluster of $\mathbf{U}^i$, and the q-th cluster of $\hat{\mathbf{U}}$, while $n_{l.}^i$, and $n_{.q}$ denote the number of objects in the l-th cluster of $\mathbf{U}^i$, and the q-th cluster of $\hat{\mathbf{U}}$, respectively. The NMI is defined between $\hat{\mathbf{U}}$, and $\mathbf{U}^i$ below in Eq. 5.11, and the ANMI is given by Eq. 5.12.

$$\mathrm{NMI}(\hat{\mathbf{U}}, \mathbf{U}^i) = \frac{\sum_{l=1}^{k_i} \sum_{q=1}^{\hat{k}} n_{lq}^i \log\left(\frac{n \times n_{lq}^i}{n_{l.}^i \times n_{.q}}\right)}{\sqrt{\left(\sum_{l=1}^{k_i} n_{l.}^i \log\frac{n_{l.}^i}{n}\right)\left(\sum_{q=1}^{\hat{k}} n_{.q} \log\frac{n_{.q}}{n}\right)}} \tag{5.11}$$

$$\mathrm{ANMI}(\hat{\mathbf{U}}; \{\mathbf{U}^i\}_{i=1}^b) = \frac{1}{b}\sum_{i=1}^{b} \mathrm{NMI}(\hat{\mathbf{U}}, \mathbf{U}^i) \tag{5.12}$$

For each experimental setting for a given dataset, the results are presented in two figures as follows. In the first figure, the results for the consensus partition with a pre-determined number of clusters are presented, where the distributions of the obtained ARI and ANMI values are shown as box-plots for ACV-k, BV-k, EAC-S, EAC-A, CSPA, HGPA, MCLA, and QMI. The second figure shows the results for the voting-based consensus algorithms, when an optimal number of clusters is estimated based on computed lifetimes. Specifically, the distributions of the ARI and ANMI for ACV versus BV are plotted, as well as pareto charts depicting the estimated $\hat{k}$ values drawn as bars in descending order of the number of times each value is estimated in 25 runs. The right vertical axis of a pareto chart shows the cumulative percentage of the total number of occurrences of $\hat{k}$. The first 95% of the cumulative distribution is displayed.

5.3.5 Results for Artificial Datasets

The results for the 2D2K dataset, where ensembles are generated with a variable number of clusters $k_i \in [6, 20]$, are shown in Fig. 5.2 for a pre-determined number of clusters $k = 2$ and in Fig. 5.3 where an estimated value for k is computed.

As observed from Figs. 5.2 and 5.3, the ACV-k and ACV algorithms extract highly accurate partitions, as indicated by the ARI values and the estimated k values. The ARI results for ACV-k and ACV are significantly higher than BV-k, EAC-S, HGPA, QMI, and BV, and they are comparable with the most accurate consensus algorithms, which in this case are EAC-A, CSPA, and MCLA. Furthermore, it is noted that estimates of k are perfect as indicated by the pareto chart in Fig. 5.3 for ACV ($k = 2$ is estimated in 100% of the runs). On the other hand, k is

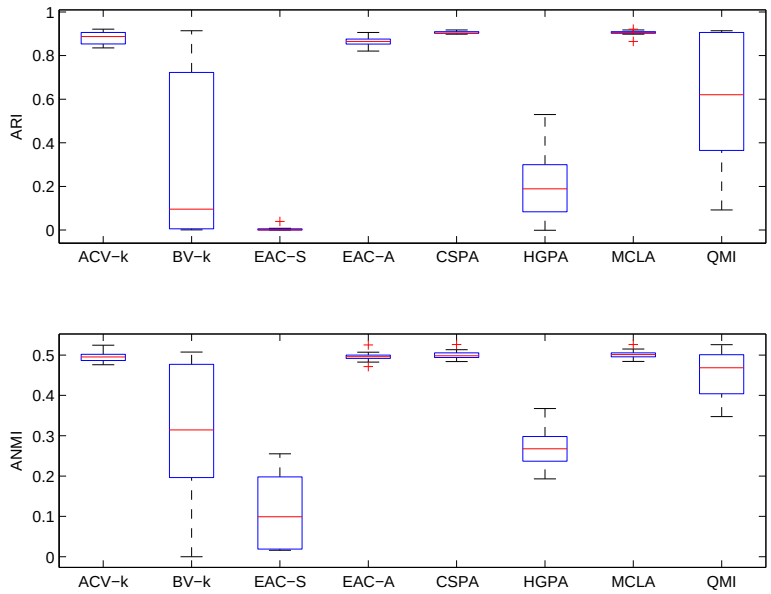

Figure 5.2: Results for the 2D2K dataset with pre-determined $k = 2$ and $k_i \in [6, 20]$

quite poorly estimated using the BV algorithm. Notably, large values for k are estimated using BV, indicating an inability to extract the global cluster structure inherent in the data. It is further noted that while the ANMI results are consistent with the ARI results in Fig. 5.2, they are inconsistent in Fig. 5.3. The high ANMI values for BV appear to be an effect of its large estimated k values. In [25], it is observed that the NMI criterion is biased toward the average number of clusters in ensemble partitions, and that maximizing it is effective only under the assumption that the number of clusters in the consensus partition is known.

Comparing the ARI results of the different consensus algorithms versus the single k-means with $k = 2$ (Table 5.1), It is noted that a few consensus algorithms (BV-k, EAC-S, HGPA, and QMI) extract a consensus partition that is significantly less accurate than that obtained using the single k-means. In other words, combining the k-means partitions with random k_i using these algorithms doesn't lead to discovering the global cluster structure of the data that the same base algorithm reveals when k is set to the true number of clusters. As for the other consensus

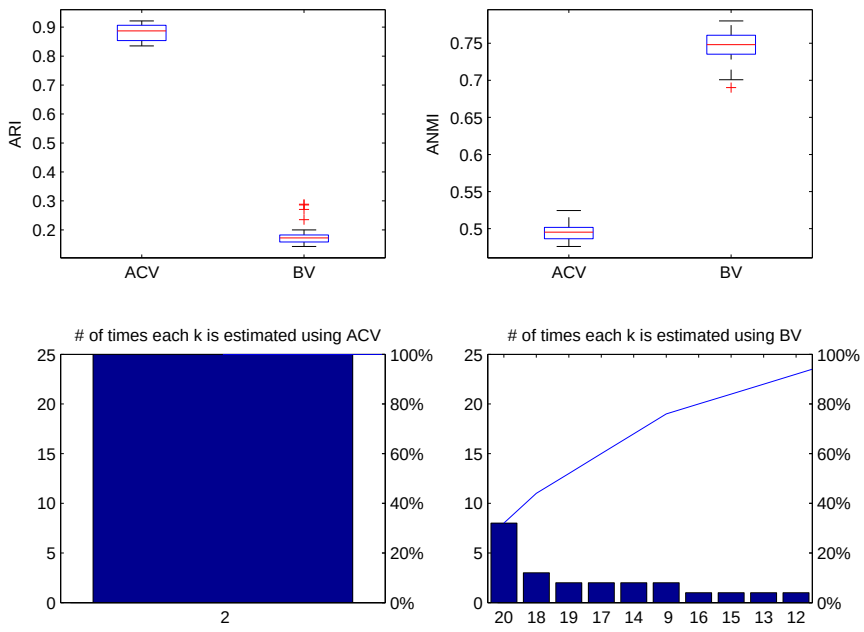

Figure 5.3: Results for the 2D2K dataset with $k_i \in [6, 20]$, where k is estimated.

algorithms (ACV-k, EAC-A, CSPA, and MCLA), they are comparable with the single k-means, which successfully extracts the cluster structure of the 2D2K dataset.

The results for the 8D5K dataset, where ensembles are generated with $k_i \in [10, 30]$, are shown in Fig. 5.4 for a pre-determined number of clusters $k = 5$ and in Fig. 5.5 where an estimated value for k is computed using each of the voting-based consensus algorithms.

For pre-determined $k = 5$, all the consensus algorithms give perfect (or almost perfect) ARI results for the 8D5K dataset. The results are more stable compared to the single k-means.

As for the results with estimated k, the BV algorithm is not stable. Estimates of k are perfect for the ACV algorithm as indicated by the pareto chart in Fig. 5.5 ($k = 5$ is estimated in 100% of the runs). On the other hand, correct $k = 5$ is estimated in approximately 50% of the runs for the BV algorithm, with other estimated values being generally quite large. The ANMI results are consistent with the ARI results, except again for BV versus ACV, as noted before.

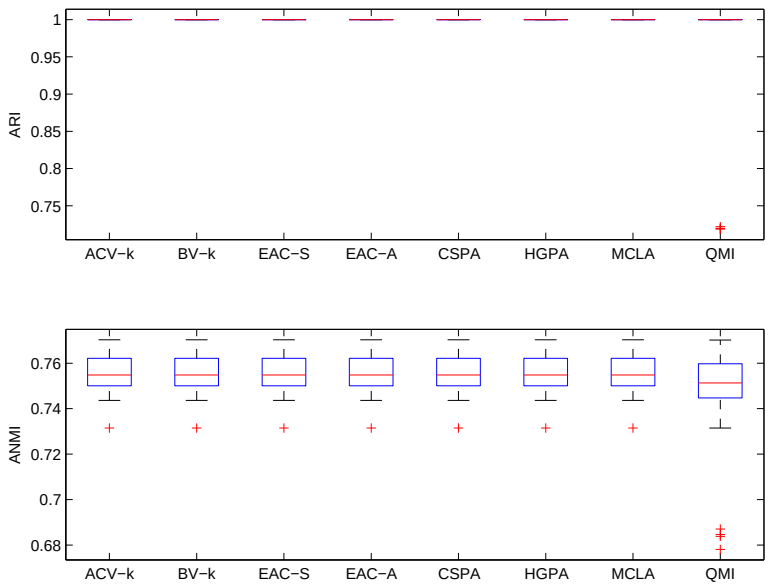

Figure 5.4: Results for the 8D5K dataset with pre-determined $k = 5$ and $k_i \in [10, 30]$

The results for the four Gauss dataset, where ensembles are generated with $k_i \in [10, 20]$, are shown in Fig. 5.6 for a pre-determined number of clusters $k = 4$ and in Fig. 5.7 where an estimated value for k is computed.

The results are similar to the 8D5k dataset. All the consensus algorithms give perfect ARI results, which are also more stable compared to the single k-means with $k = 4$. However, it noted that unlike BV-k, the ARI results for BV are not stable, and generally much less accurate. Estimates of k using the ACV algorithm are perfect as indicated by the pareto chart in Fig. 5.7 ($k = 4$ is estimated in 100% of the runs), whereas correct $k = 4$ is estimated in approximately 40% of the runs for the BV algorithm, with remaining estimated values being relatively large. The same observation about the ANMI results is noted, which is consistent with the previous datasets.

The results for the easy doughnut dataset, where ensembles are generated with $k_i \in [6, 12]$, are shown in Fig. 5.8 for a pre-determined number of clusters $k = 2$ and in Fig. 5.9 where an

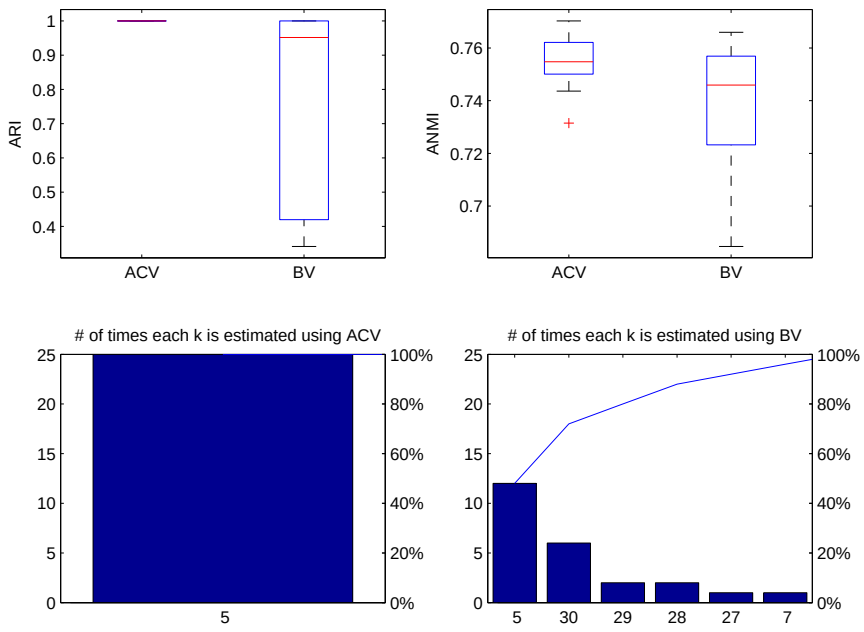

Figure 5.5: Results for the 8D5K dataset with $k_i \in [10, 30]$, where k is estimated.

estimated value for k is computed.

The dataset has a spiral cluster structure that the k-means algorithm is not capable of discovering. This k-means failure is reflected by the ARI values in Table 5.1.

As observed in Fig. 5.8, all consensus algorithms, except BV-k, lead to consensus partitions with perfect accuracy as indicated by the ARI results for the easy doughnut dataset. Furthermore, estimates of k are perfect for the ACV algorithm, where $k = 2$ is estimated in 96% of the runs, as observed in Fig. 5.9. However, the true $k = 2$ is never correctly estimated using the BV algorithm, and consensus partitions with low-accuracy are obtained. Estimated values of k are again relatively large, and the same observation about the ANMI results is noted, which is again consistent with the previous datasets.

The results for the difficult doughnut dataset, where ensembles are generated with $k_i \in [6, 12]$, are shown in Fig. 5.10 for a pre-determined number of clusters $k = 2$ and in Fig. 5.11 where an

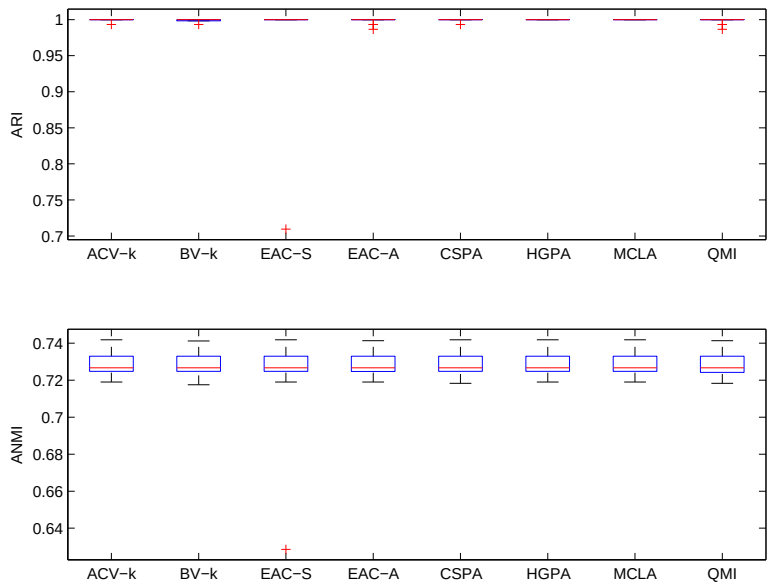

Figure 5.6: Results for the four Gauss dataset with pre-determined $k = 4$ and $k_i \in [10, 20]$

estimated value for k is computed.

The dataset has a spiral but more difficult cluster structure than the previous dataset due to less separation between the clusters. The k-means algorithm fails in discovering the true partition, as reflected by the ARI values in Table 5.1.

As observed in Fig. 5.10, all consensus algorithms, except BV-k, lead to consensus partitions with very high accuracy. Furthermore, estimated partitions using the ACV algorithm are considerably more accurate compared to the BV algorithm. Correct estimates of k for the ACV algorithm are achieved in 48% of the runs, which is a lower percentage compared to other datasets with easier cluster structures. One the other hand, for the BV algorithm, it is noted that in none of the runs was the true $k = 2$ correctly estimated.

Figures 5.12 and 5.13 show the results for the two Gauss dataset where ensembles are generated with a variable number of clusters $k_i \in [8, 16]$, when a pre-determined number of clusters $k = 2$ is given, and when an estimated value for k is sought, respectively.

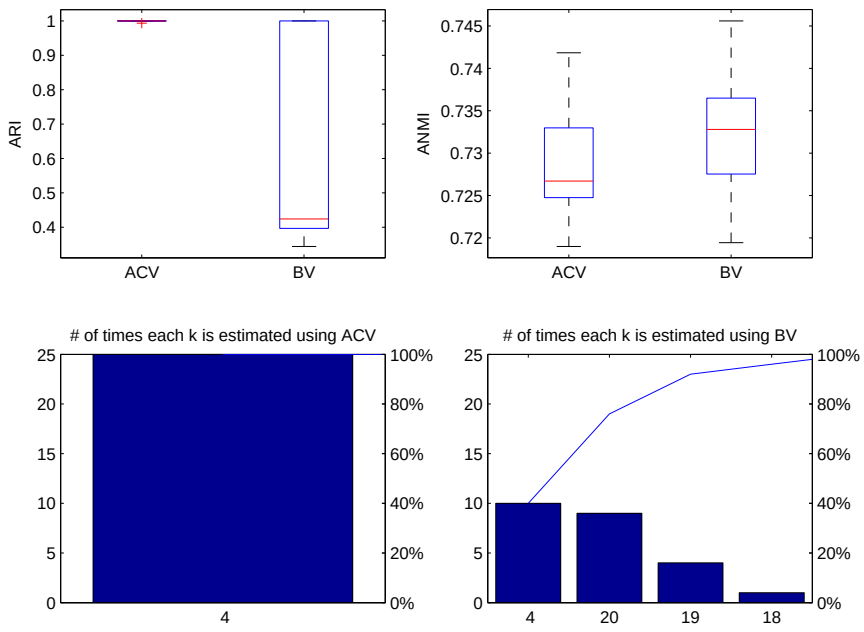

Figure 5.7: Results for the four Gauss dataset with $k_i \in [10, 20]$, where k is estimated.

The consensus partitions obtained using the ACV-k algorithm are significantly more accurate compared to all other consensus algorithms. On the other hand, BV-k performs quite poorly in this case. The true cluster structure of the two Gauss dataset is relatively difficult to extract. It is noted that correct estimates of k for the ACV algorithm are achieved in 52% of the runs. For the BV algorithm, estimated values of k are again generally large, and in none of runs was the true $k = 2$ correctly estimated.

5.3.6 Results for Real Datasets

The results for the breast cancer dataset, where ensembles are generated with $k_i \in [6, 12]$, are presented in Fig. 5.14 for a pre-determined number of clusters $k = 2$ and in Fig. 5.15 where an estimated value for k is computed. Furthermore, results for ensembles with $k_i = 15$, $\forall i$ are shown in Figs. 5.16 and 5.17.

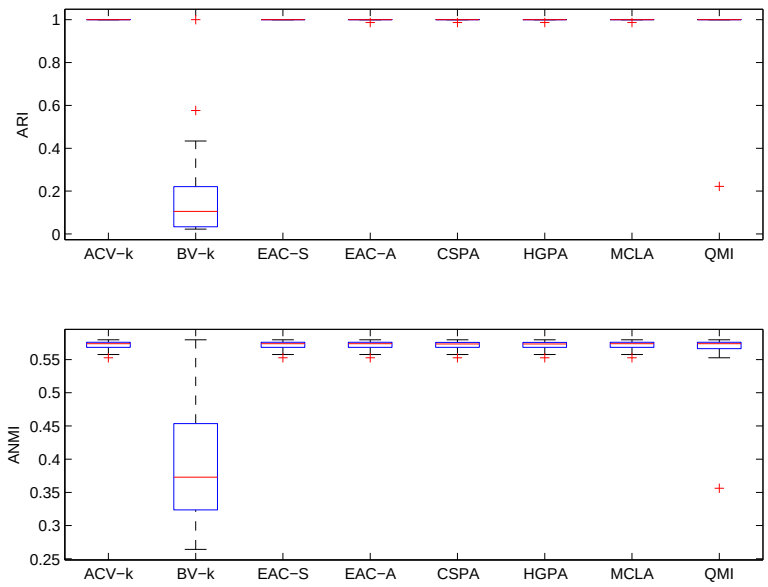

Figure 5.8: Results for the easy doughnut dataset with pre-determined $k = 2$ and $k_i \in [6, 12]$

As observed in Fig. 5.14, the quality of the consensus partitions obtained using ACV-k and ACV is better than that obtained using all other consensus algorithms, where the improvement is substantial in most cases. Furthermore, the true number of clusters is correctly estimated as observed in Fig. 5.15. On the other hand, BV and BV-k perform quite poorly.

Moreover, as observed in Figs. 5.16 and 5.17, the consensus partitions obtained using ACV and ACV-k are as accurate and robust as EAC-A and substantially better than all other consensus algorithms. The estimated number of clusters are also quite accurate, whereas BV and BV-k perform relatively poorly. Notably, the number of estimated clusters using BV is almost always equal to k_i. This result indicates that when k_i is fixed, the bipartite matching scheme causes the aggregation to be tied to this value, unlike the cumulative voting scheme, which is more effective in enabling the global cluster structures to be discovered.

The results for the optical digits dataset, where ensembles are generated with $k_i \in [15, 30]$, are presented in Fig. 5.18 for a pre-determined number of clusters $k = 10$ and in Fig. 5.19 where

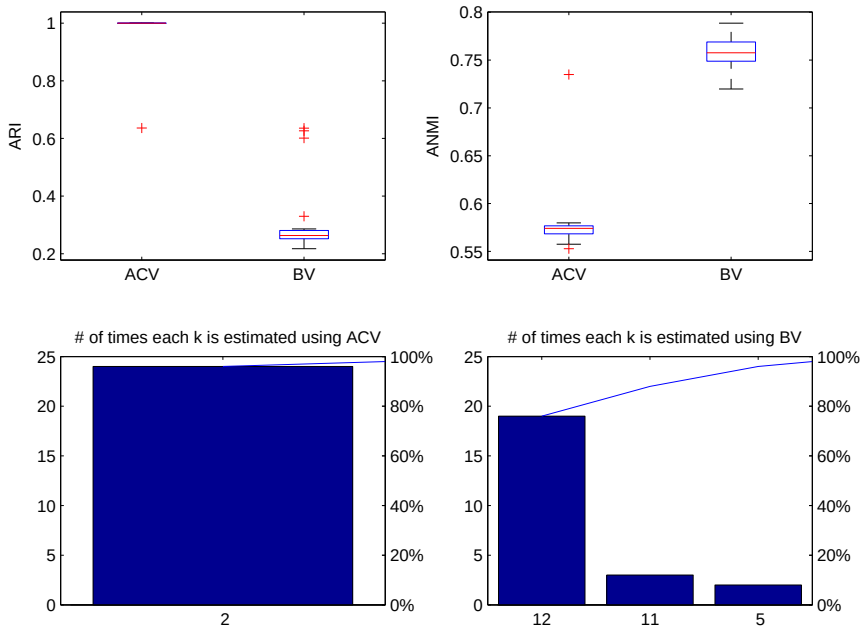

Figure 5.9: Results for the easy doughnut dataset with $k_i \in [6, 12]$, where k is estimated.

an estimated value for k is computed. Furthermore, results for ensembles with $k_i = 30$, $\forall i$ are shown in Figs. 5.20 and 5.21.

As observed in Fig. 5.18, the quality of the consensus partitions obtained using ACV-k is comparable with the best performing consensus algorithms and is better than the single k-means with $k = 10$ (Table 5.1). Furthermore, the estimated number of clusters are close to 10 using ACV, where also relatively accurate consensus partitions are estimated as noted from the ARI values in Fig. 5.19. On the other hand, BV and BV-k are less accurate and estimated cluster structure is quite fragmented as noted from the large estimates for the number of clusters.

Furthermore, as observed in Figs. 5.20 and 5.21, the results are consistent with the case of ensembles with a variable number of clusters. It is again noted that BV leads only to solutions with estimated number of clusters equals to k_i (which is 30 in this case). These results demonstrate the limitations of the bipartite matching scheme with this type of ensemble.

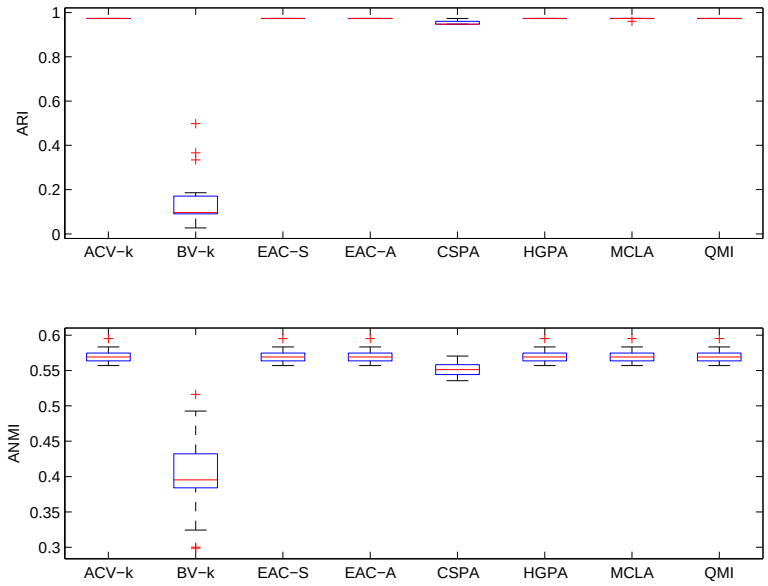

Figure 5.10: Results for the difficult doughnut dataset with pre-determined $k = 2$ and $k_i \in [6, 12]$

The results for the yahoo! dataset, where ensembles are generated with $k_i \in [12, 24]$, are presented in Fig. 5.22 for a pre-determined number of clusters $k = 6$ and in Fig. 5.23 where an estimated value for k is computed. Furthermore, results for ensembles with $k_i = 24$, $\forall i$ are shown in Figs. 5.24 and 5.25.

It is noted that for the yahoo! dataset, only the $O(n)$ algorithms are applied, whereas the $O(n^2)$ consensus algorithms are excluded as they are computationally burdensome for large datasets. Furthermore, previous results on smaller datasets do not suggest that significant gains in accuracy may be anticipated compared to ACV-k. It is noted that ACV-k is generally competitive compared to EAC-A, which is one of the most consistently well-performing algorithms.

As observed in Fig. 5.22, the accuracy of the obtained consensus partitions using ACV-k is substantially better than that obtained using all other consensus algorithms, as indicated by the ARI measure. The ACV-k algorithm also achieves significant accuracy gains over the single k-means algorithm.

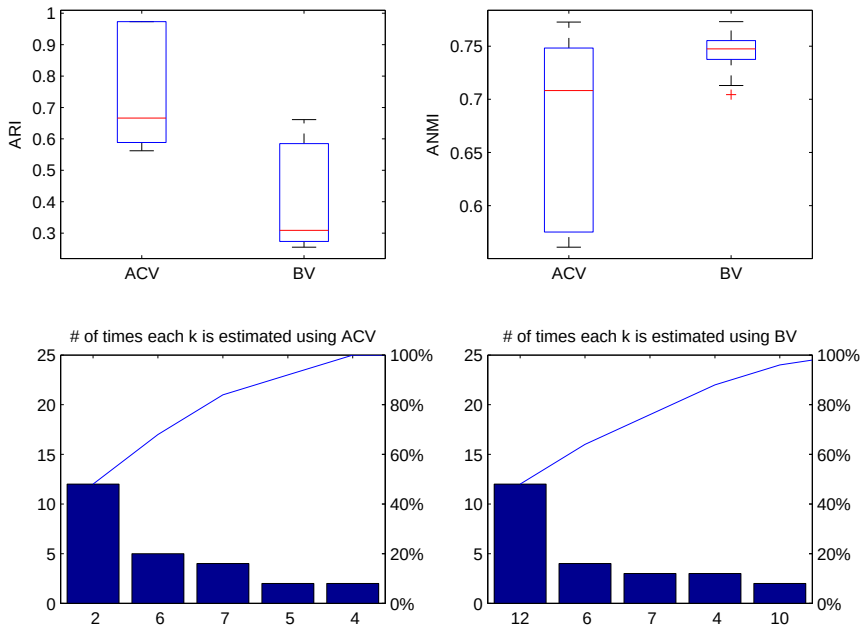

Figure 5.11: Results for the difficult doughnut dataset with $k_i \in [6, 12]$, where k is estimated.

As for the ACV algorithm, gains in accuracy are still achieved in the estimated consensus partition as indicated in Fig. 5.24. However, the exact number of clusters is quite difficult to determine for the yahoo! dataset. Estimated values using ACV reflect that the global cluster structure is detected to some extent, whereas the results using BV appear to be poor as indicated by observing the ARI values and the estimated number of clusters.

Furthermore, the results in Fig. 5.24 show that the performance of the ACV-k algorithm exceeds all other consensus algorithms. Estimated values using ACV show that reasonably accurate results are achievable, whereas the BV algorithm leads to a substantially less accurate solution with the estimated number of consensus clusters exclusively limited to $k = 24$ clusters.

5.4 Summary

In this section, a summary of the empirical study and the conclusions are presented.

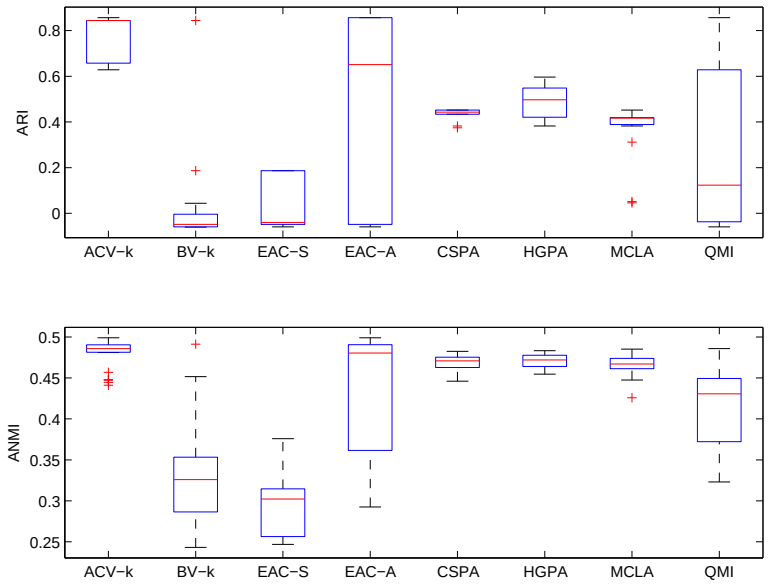

Figure 5.12: Results for the two Gauss dataset with pre-determined $k = 2$ and $k_i \in [8, 16]$

5.4.1 Results Summary

Table 5.2 summarizes the empirical results for all the experiments reported in this chapter. The summary is based on the accuracy of the consensus partitions as measured by the ARI. The table highlights the winners between the voting-based consensus algorithms ACV versus BV, ACV-k versus BV-k, as well as the overall winner among all consensus algorithms for pre-determined k, including ACV-k and BV-k. If several consensus algorithms have a comparable performance (as indicated by their corresponding boxplots), a list of winners is given.

For all the ensembles generated in the empirical study, it is noted that each of the proposed ACV and ACV-k algorithms wins over BV and BV-k, respectively, almost always. Furthermore, the ACV-k is always the overall winner or one of the overall winners, as noted in Table 5.2.

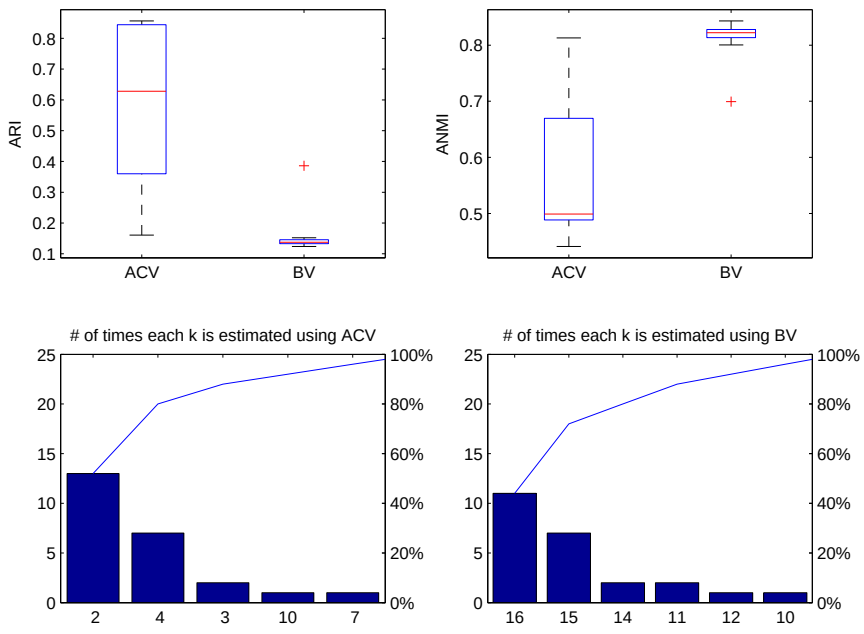

Figure 5.13: Results for the two Gauss dataset with $k_i \in [6, 18]$, where k is estimated.

5.4.2 Conclusion

In this chapter, partition ensembles were generated using the k-means algorithm, where the number of clusters k_i per ensemble partition is larger than desired or anticipated for the consensus partition, and also where it is randomly selected. The aggregated partition using the cumulative voting and bipartite matching schemes is viewed as an aggregated distributional representation for the ensemble. An efficient approximation algorithm is developed from an information-theoretic basis to search for the most compressed summary of the aggregated distribution that preserves the maximum amount of information and to reveal a global and cohesive consensus partition. The information theoretic algorithm, referred to as `JS-Alink`, is applied in conjunction with each of the `Ada-cVote` and the `bVote` algorithms.

Furthermore, an approach to estimating an optimal number of clusters for the data was applied. The approach is based on the idea of a cluster lifetime as measured from a generated

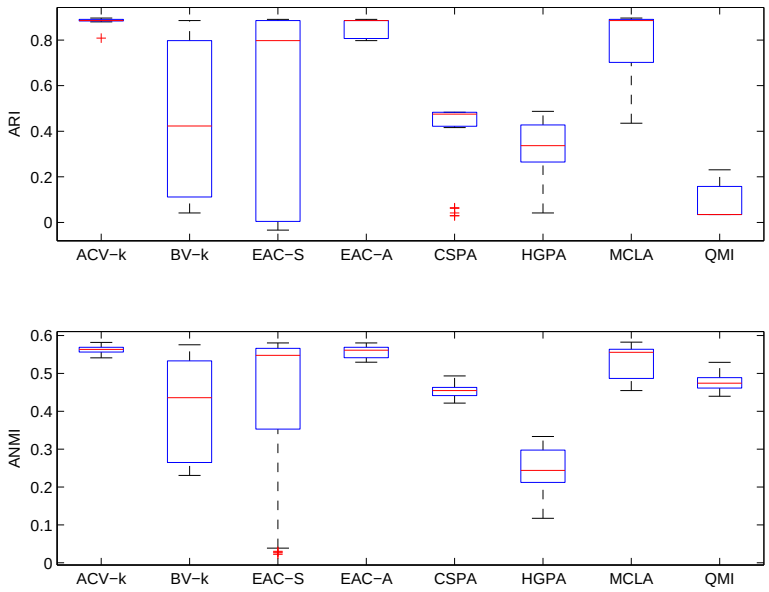

Figure 5.14: Results for the breast cancer dataset with pre-determined $k = 2$ and $k_i \in [6, 12]$

hierarchy of k-partitions. The application of each of the aggregation algorithms `Ada-cVote` and `bVote` with the `JS-Alink`, in addition to whether the approach to estimating k is applied or a pre-determined k is given, lead to defining the consensus algorithms abbreviated as ACV-k, ACV, BV-k, and BV.

An empirical study was conducted to validate the proposed algorithms. A comparative evaluation of the different voting schemes and several recent consensus algorithms was presented. A number of artificial and real-world datasets with different levels and types of difficulties were used in the study. Experimental results demonstrate that the cumulative voting scheme is substantially more suitable than the bipartite matching for the types of cluster ensembles considered here, when used in conjunction with the `JS-Alink` algorithm. Furthermore, the results of the ACV-k algorithm were either comparable to or better than other recent consensus algorithms. In some case, substantial improvements were achieved over other consensus algorithms. On the other hand, BV-k performed poorly, in general.

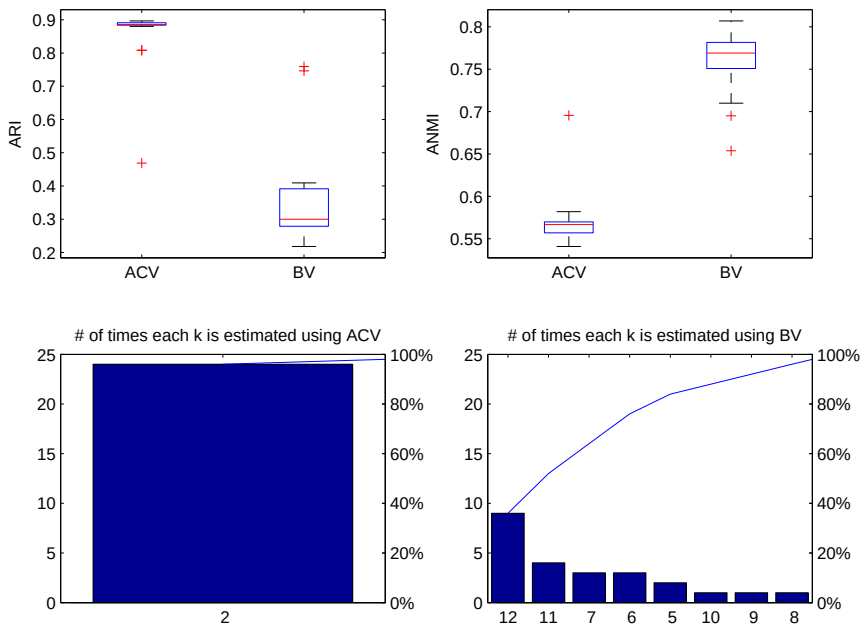

Figure 5.15: Results for the breast cancer dataset with $k_i \in [6, 12]$, where k is estimated.

Moreover, it was demonstrated that accurate estimates of the true number of clusters can be often achieved using ACV. Poor estimates of the number of clusters are consistently observed when the bipartite matching scheme is applied. The estimated number of clusters using BV are generally large, and the global cluster structure of the data is not revealed. In the case of ensembles with a large but fixed k_i, it is consistently noted that BV leads to solutions with the same number of clusters $k = k_i$. In this case, the BV algorithm does not lead to solutions beyond that determined by the individual ensemble partitions.

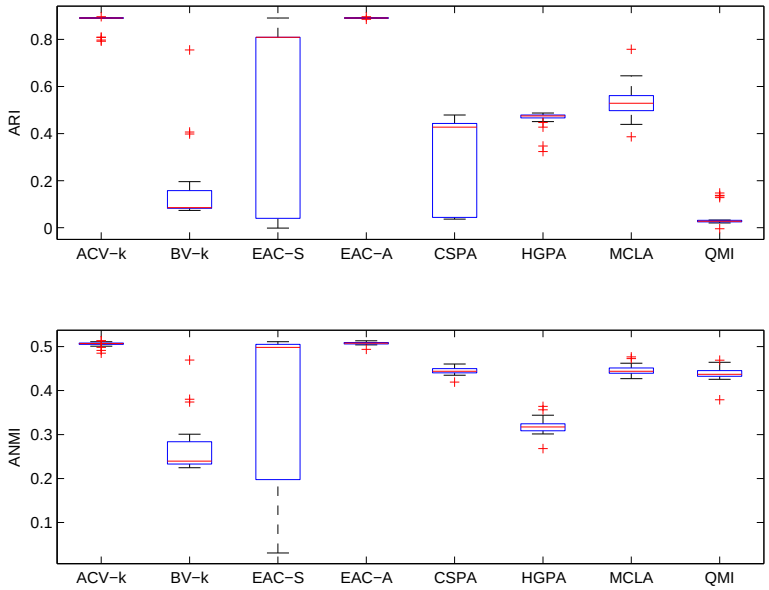

Figure 5.16: Results for the breast cancer dataset with pre-determined $k = 2$ and $k_i = 15$

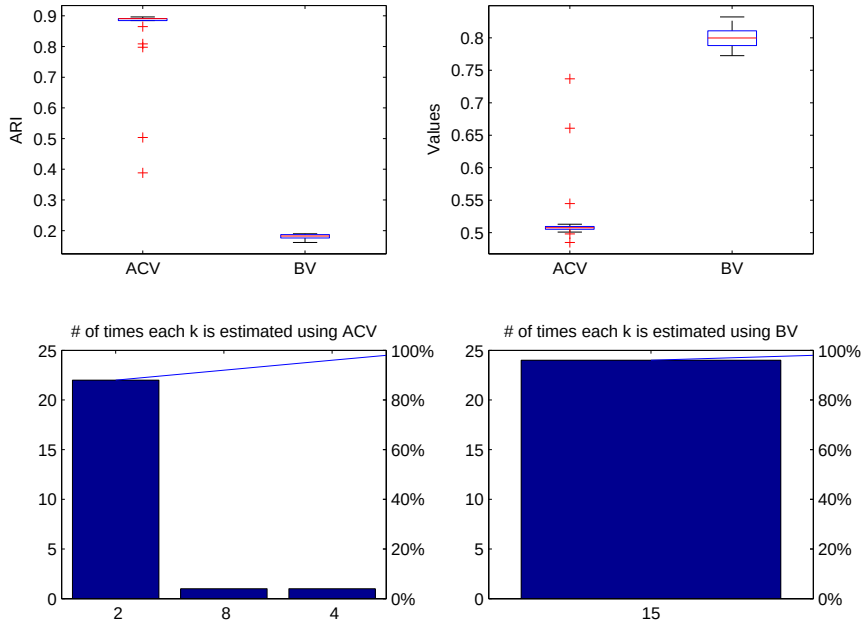

Figure 5.17: Results for the breast cancer dataset with $k_i = 15$, where k is estimated.

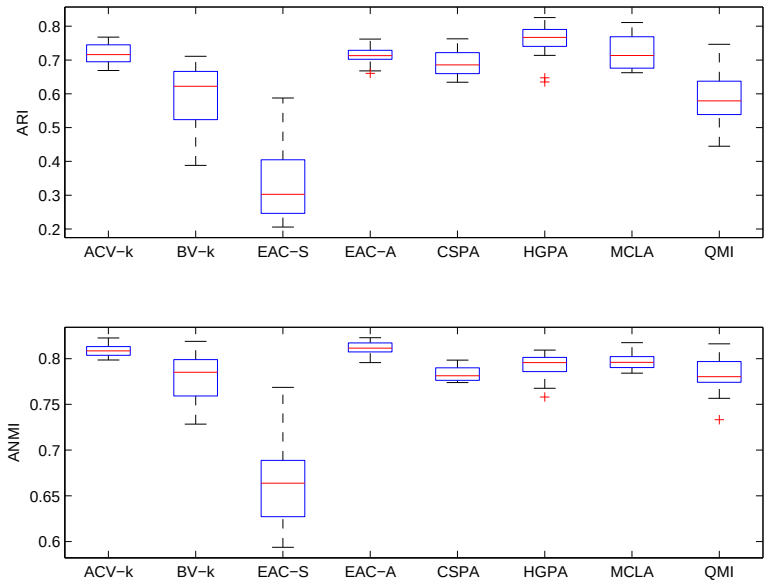

Figure 5.18: Results for the optical digits dataset with pre-determined $k = 10$ and $k_i \in [15, 30]$

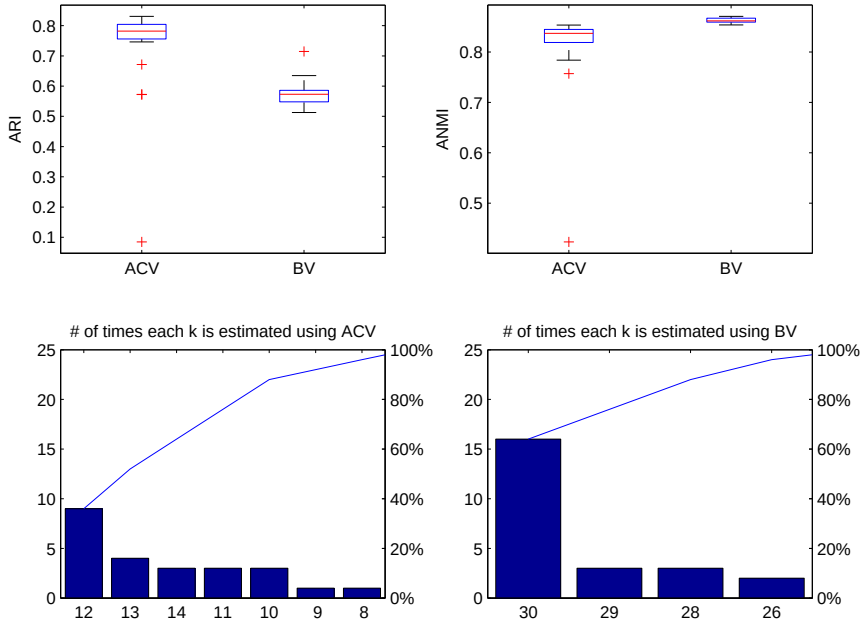

Figure 5.19: Results for the optical digits dataset with $k_i \in [15, 30]$, where k is estimated.

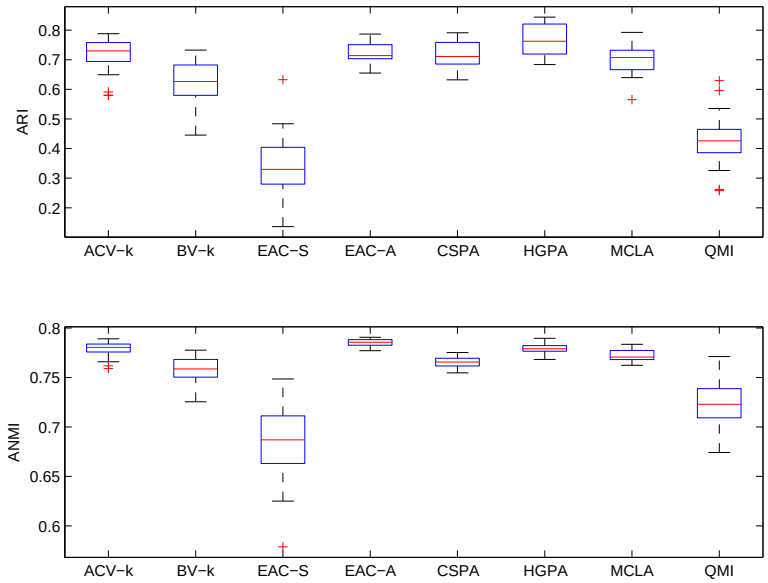

Figure 5.20: Results for the optical digits dataset with pre-determined $k = 10$ and $k_i = 30$

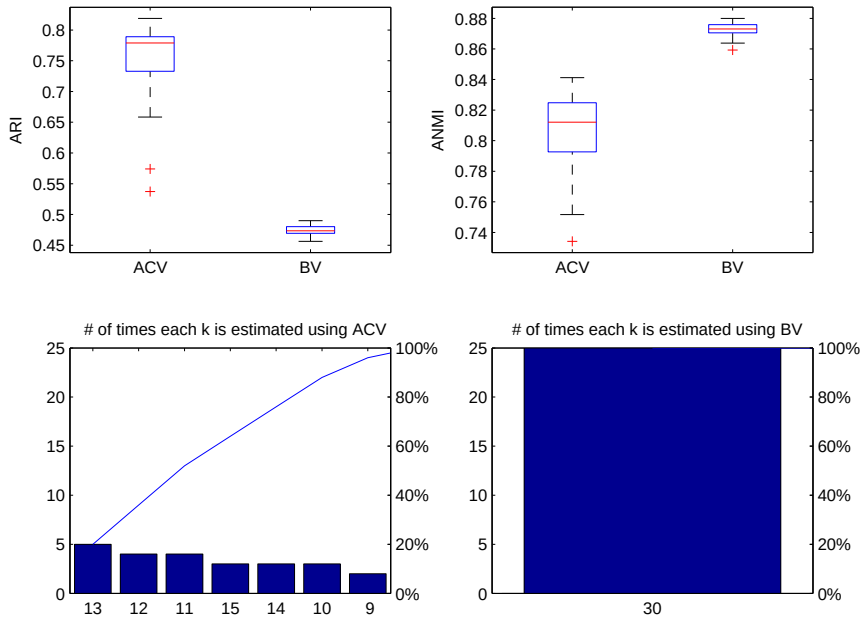

Figure 5.21: Results for the optical digits dataset with $k_i = 30$, where k is estimated.

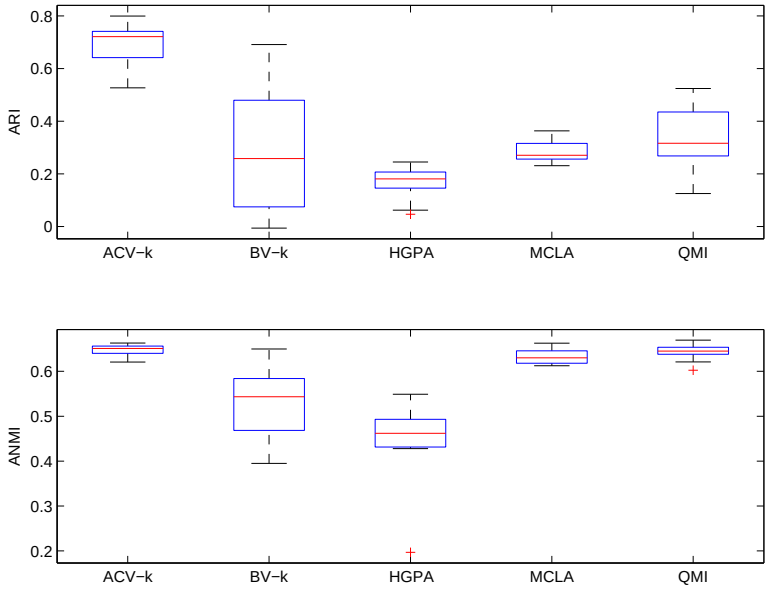

Figure 5.22: Results for the Yahoo! dataset with pre-determined $k = 6$ and $k_i \in [12, 24]$

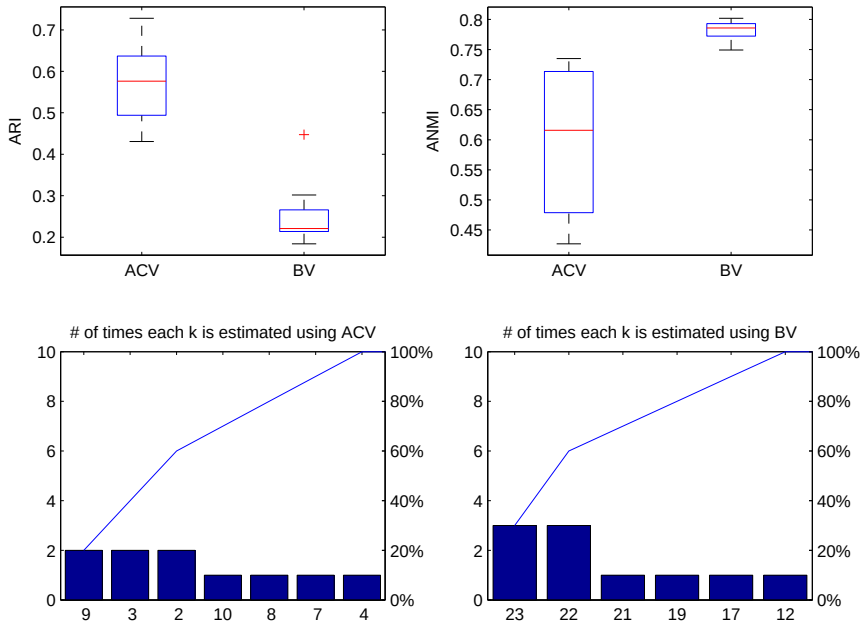

Figure 5.23: Results for the Yahoo! dataset with $k_i \in [12, 24]$, where k is estimated.

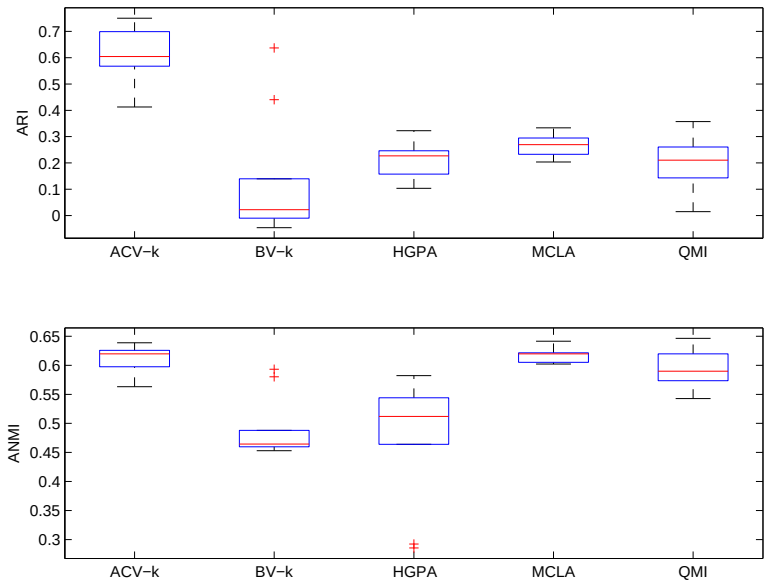

Figure 5.24: Results for the Yahoo! dataset with pre-determined $k = 6$ and $k_i = 24,\ \forall i$

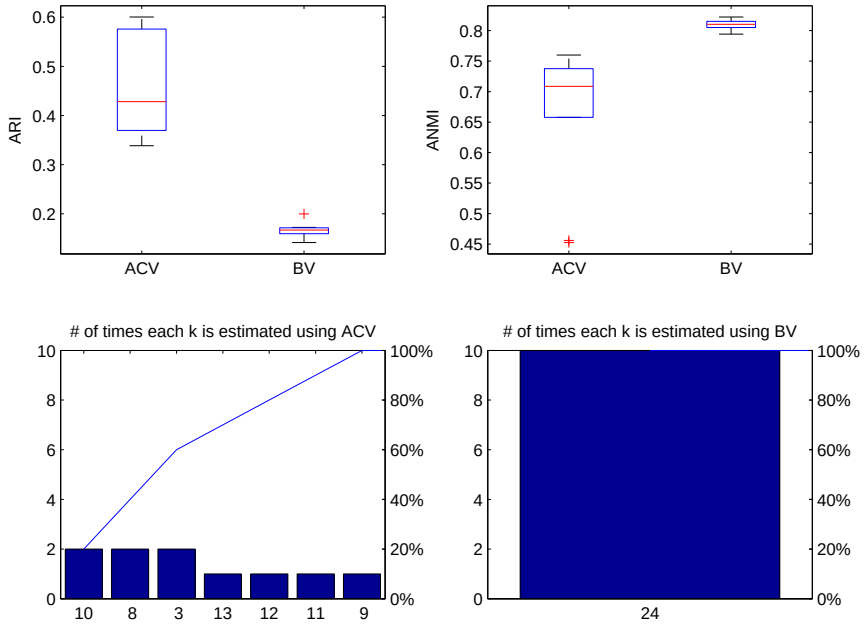

Figure 5.25: Results for the Yahoo! dataset with $k_i = 24,\ \forall i$, where k is estimated.

Table 5.2: Summary of experimental results based on the ARI measure.

Dataset	$[k_{\min}, k_{\max}]$	**ACV/BV Winner**	**ACV-k/BV-k Winner**	**Overall Winner**
2D2K	$[6, 20]$	ACV	ACV-k	CSPA, MCLA, EAC-A, ACV-k
8D5K	$[10, 30]$	ACV	ACV-k, BV-k	All
Four Gauss	$[10, 20]$	ACV	ACV-k, BV-k	All
Easy doughnut	$[6, 12]$	ACV	ACV-k	All except BV-k
Difficult doughnut	$[6, 12]$	ACV	ACV-k	All except BV-k
Two Gauss	$[6, 18]$	ACV	ACV-k	ACV-k
Breast cancer	$[6, 12]$	ACV	ACV-k	ACV-k, EAC-A
Breast cancer	$[15, 15]$	ACV	ACV-k	ACV-k, EAC-A
Optical digits	$[15, 30]$	ACV	ACV-k	HGPA, EAC-A, MCLA, ACV-k, CSPA
Optical digits	$[30, 30]$	ACV	ACV-k	HGPA, EAC-A, CSPA, ACV-k, MCLA
Yahoo!	$[12, 24]$	ACV	ACV-k	ACV-k
Yahoo!	$[24, 24]$	ACV	ACV-k	ACV-k

Chapter 6

Conclusions

6.1 The Voting-Based Consensus Problem

The basic goal of reconciling an ensemble of partitions is to obtain a consensus partition that optimally summarizes an ensemble. For consensus partitions to be useful, they should reveal cluster structures for the data with improved accuracy and stability compared to the individual ensemble partitions. As demonstrated in this dissertation, the quality of the obtained consensus solutions highly depends on the ensemble generation mechanism and on the suitability of the consensus method for effectively combining the generated ensemble. Hence, it is important to examine the effectiveness of consensus methods against different partition generation techniques.

Typically, consensus methods for partitions derive an ensemble representation that is subsequently used as the basis for extracting an optimal consensus partition. In most cases, the derived representation sidesteps the relabeling problem. On the other hand, voting-based consensus methods [28, 31, 33, 39, 42, 43] represent a distinct class of consensus methods, whereby direct parallels with the aggregation of supervised learners [17, 18, 21] are sought. Specifically, voting-based methods derive an ensemble representation consisting of a central aggregated partition, by directly addressing the ensemble relabeling problem.

Unlike the aggregation of supervised classifications, the voting-based aggregation of partitions requires the simultaneous optimization of relabeling the ensemble partitions with respect to the sought (aggregated) partition and of the aggregated partition with respect to the relabeled ensemble partitions [33]. The optimal relabeling of multiple partitions corresponds to a multi-dimensional assignment problem (MAP), which unlike the pairwise relabeling problem, is NP-hard [34]. Several efficient algorithms are proposed in recent work [28, 31, 33, 39, 42, 43].

An important element of the voting-based aggregation problem is the pairwise relabeling of an ensemble partition with respect to a representative partition. The pairwise relabeling is referred to as the voting problem. The voting problem is commonly formulated as a weighted bipartite matching problem [28, 31, 33, 42, 43], which is a combinatorial optimization problem, that is also known as the linear sum assignment problem. Based on this formulation, one looks for an optimal cluster label permutation for each ensemble partition such that a constrained loss with respect to a representative partition is minimized. The general measure for the relabeling loss is the mean squared error, which is equivalent to the probability of error in the case of hard ensembles, and to the misclassifcation rate, in the case of a hard representative partition. The solution is obtained using Kuhn's Hungarian method, which is $O(k^3)$. In the case of unequal numbers of clusters, empty clusters are added to the partition with fewer clusters.

6.2 Contributions

A summary of the contributions is presented in this section.

6.2.1 A New Formulation for the Voting Problem

A general formulation for the voting problem as a multi-response regression problem was introduced and the cumulative voting scheme was proposed as a special instance that corresponds to fitting a linear model by least squares estimation. Due to the additional constraints in the bipartite matching formulation of the voting problem, the achievable loss based on bipartite matching

is bounded from below by the achievable loss based on cumulative voting. The general formulation offers more flexibility in defining voting schemes that can be applied to model substantial variability between partitions, such as a variable number of clusters.

For the ensemble aggregation, a general iterative algorithm (`Vote`) was applied, with variants corresponding to cumulative voting (`cVote`) and bipartite matching (`bVote`). The convergence properties of the aggregated partition based on the bipartite matching scheme in conjunction with plurality voting was formally established in [43] for a particular partition generation model. A simulation-based analysis was presented. It demonstrated that `bVote` is more suitable than `cVote` for this model, which corresponds to uniform partitions where each is generated as a noisy permutation of an underlying labeling, according to a probability of error. For other types of generation models, such as partitions with a variable number of clusters, the aggregated partition was viewed as a distributional representation and define a criterion for extracting an optimally compressed consensus partition based on the estimated aggregated distribution.

It is noted that the aggregated solution using the `Vote` algorithm depends on the initially selected reference partition and on the order in which the ensemble partitions are aggregated. However, because of the suitability of `bVote` to ensembles of uniform partitions, the aggregated partitions are stable and they converge to the underlying labeling used to generate the ensemble. On the other hand, the aggregated solutions using `cVote` are unstable, when the probability of error increases, indicating its unsuitability for this type of ensemble.

6.2.2 A study of the Properties of Cumulative Voting

Relation to Co-Association Based Consensus

The properties of the proposed cumulative voting scheme were investigated. In particular, the relationship between cumulative voting and the co-association matrix was derived. The relationship is outlined by defining an un-normalized cumulative voting scheme. A fixed-reference aggregation algorithm referred to as `URef-cVote` is developed in conjunction with the un-normalized scheme.

In the special case where the reference partition corresponds to the partition of n singleton clusters represented by the n identity matrix $\mathbf{I}_n$, the aggregated partition computed using `URef-cVote` is the co-association matrix. For the un-normalized cumulative voting scheme, the underlying loss is a least squares objective function with a constraint on the estimated co-occurrence values of the objects and the representative clusters to sum up to the size of the voting clusters.

Adaptive Aggregation

A notable property of the cumulative voting scheme is that when the data objects are assumed to be sampled uniformly at random, the class distribution associated with the reference partition are preserved in the relabeled ensemble partitions as well as the aggregated partition. Based on this property, a criterion was defined for selecting the initial reference and the aggregation sequence of the ensemble partition so as to minimize the loss of mutual information associated with the estimated aggregated distribution. The `Ada-cVote` algorithm was developed as an adaptive aggregation algorithm for the cumulative voting scheme. Instead of considering the ensemble partitions in a random order as in the case of the `Vote` algorithm, `Ada-cVote` aims at selecting the initial reference partition according to the defined criterion. An important feature of the `Ada-cVote` algorithm is that the obtained aggregated partition is invariant to the order of the ensemble partitions and the initial reference, unlike `cVote`. Experimental evidence was presented, showing that the adaptivity feature is only effective with `cVote`, but not when applied with `bVote`.

6.2.3 Compression of Aggregated Representation

Efficient Algorithm

The ensemble generation mechanisms proposed in [25, 36] were applied, where for each ensemble partition, the number of clusters k_i is larger than desired or anticipated, or when it is randomly selected. For this type of ensemble, a principled information theoretic approach was proposed

for extracting a consensus partition that represents a global and cohesive cluster structure for the data. The `JS-Alink` algorithm was developed as an efficient agglomerative algorithm that is based on the information bottleneck formulation of Tishby et al. [8]. The algorithm minimizes the Jensen-Shannon divergence within the merged cluster, or equivalently, it minimizes the loss in mutual information associated the distributional representation of the data that is due to compressing the representation. Furthermore, an approach was applied for estimating an optimal number of clusters based on the idea of a cluster lifetime proposed in [25], which is measured from a generated hierarchy of k-partitions.

Empirical Validation

Several artificial and real-world datasets were used. They are characterized by various challenges, including a large text data with very high dimensionality. Consistent evidence demonstrates that the cumulative voting scheme is substantially more suitable for this type of ensemble than the bipartite matching scheme. Furthermore, the consensus partitions obtained using the proposed consensus method are either comparable with or better than those obtained using several recent consensus algorithms. Moreover, accurate estimates of the true number of clusters are often achieved using the cumulative voting scheme, whereas consistently poor estimates are achieved based on bipartite matching. The results provide consistent evidence on the unsuitability of the bipartite matching scheme for this type of ensemble.

6.2.4 Computational Efficiency

Overall, the proposed voting-based consensus method, implemented by the ACV and ACV-k algorithms, is computationally efficient. The $O(n^2)$ complexity is avoided at each stage. Furthermore, competitively accurate and stable results are consistently achieved consistently, compared to other consensus methods, for ensembles with randomly selected number of clusters. Thus, the proposed consensus method offers a computationally more efficient alternative for co-association

based methods. The computational complexity is $O(k^2nb)$, where $k \ll n$. The bipartite matching based consensus method is $O(k^3nb)$. Table **??** summarizes the computational complexity of the different consensus algorithms. The HGPA is $O(nkb)$, MCLA is $O(nk^2b^2)$, QMI is $O(nkb)$, co-association-based consensus functions with single and average link are $O(n^2b)$, and the graph-based CSPA algorithm is $O(n^2kb)$ [25, 38]

6.3 Future Work

This thesis opens several future research directions for voting-based consensus clustering. A few interesting directions are highlighted in this section.

6.3.1 Multi-Response Regression Formulation

The regression formulation introduced in this thesis for the voting problem can be further developed to create and analyze new voting schemes. In the cumulative voting scheme, it is assumed that a linear model is suitable for describing the relationship between each ensemble partition and the reference partition. The form of the regression function that underlies the relationship between the input and output variables can be further investigated by exploring other models.

6.3.2 Application in Bioinformatics

DNA micro-array technology provides the means for measuring the expression levels of tens of thousands of genes, simultaneously, for a given experimental sample. It has the potential to help further the understanding of biological processes and to introduce important applications in pharmaceutical and clinical research [85]. Gene expression data is usually represented as a data matrix of genes versus samples, where each entry represents the expression level of a gene for a given sample. Data clustering of gene expression matrices aims at finding relevant groupings, such as groups of genes with similar functionality, and at extracting gene structures and biologically meaningful information.

The difficulties of the data clustering task for the micro-arrays makes the application of cluster ensemble methods particularly interesting. In fact, similarity-based consensus algorithms as well as voting algorithms based on bipartite matching have been successfully applied in on gene expression micro-array data [31,40]. Hence, the extension of the voting-based consensus clustering algorithms proposed in this thesis to the cluster analysis of bioinformatics data represents an interesting future direction.

6.3.3 Application to Model-Based Cluster Ensembles

Finite mixture models represent a principled statistical approach to cluster analysis [5]. In model-based clustering, data are represented by a mixture model, where each component probability distribution in the mixture corresponds to a cluster. The EM algorithm for maximum likelihood is applied to determine a partition solution [75, 76]. The proposed framework and the introduced cumulative voting-based aggregation can be applied for estimating the cluster conditional probability distribution and a consensus partition for a model-based cluster ensemble. Such application represents an interesting future direction.

6.3.4 Consensus Clustering Validation

The accuracy and robustness of the computed voting-based consensus partitions depend on the input partitions and the effectiveness of the voting scheme in leading to a relevant consensus clustering. Several design issues are at play, including the selection of the base clustering algorithm and the setup of the ensemble parameters, such as the number of clusters k_i of the ensemble partitions and the ensemble size. For improper values of the ensemble parameters, the consensus method leads to a clustering solution that is inadequate for the dataset. An in-depth investigation of validation methods for the extracted consensus cluster structures is an important future research direction. The validation seeks to evaluates the consensus tendency in order to determine values of the design parameters that best fits a particular dataset.

Bibliography

[1] J.A. Hartigan. *Clustering Algorithms*. Wiley, 1975.

[2] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[3] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data*. Wiley, 1990.

[4] A.K. Jain, M.N Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, September 1999.

[5] C. Fraley and A. E. Raftery. Model-based clustering, discriminant analysis, and density estimation. Technical report, University of Washington, October 2000.

[6] J. M. Buhmann. Data clustering and learning. In M. Arbib, editor, *Handbook of Brain Theory and Neural Networks*. MIT Press., 2002.

[7] J. Kleinberg. An impossibility theorem for clustering. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2002.

[8] N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[9] Elena D. Cristofor. *Information-Theoretic Methods In Clustering*. PhD thesis, University of Massachusetts, 2002.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data mining, Inference and Prediction*. Springer, 2001.

[11] B. G. Mirkin. On the problem of reconciling partitions. In H M Blalock, A. Aganbegian, F. M. Borodkin, R. Boudon, and V Capecchi, editors, *Quatitative Sociology: International Perspectives on Mathematical and Statistical Modeling*, Quantitative Studies in Social Relations, pages 441–449. Academic Press, 1975.

[12] S. Régnier. Etudes sur le polyèdre des partitions. *Mathmatiques et Sciences Humaines*, 82:85–111, 1983.

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

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

[15] J.-P. Barthlemy and B. Leclerc. The median procedure for partitions. In *Partitioning Data Sets*, volume 19 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 3–33, 1995.

[16] William H. E. Day and F. R. McMorris. *Axiomatic Consensus Theory in Group Choice and Biomathematics*, volume 39 of *SIAM Frontier in Applied Mathematics*. Society for Industrial and Applied Mathematics, 2003.

[17] Leo Breiman. Bagging predictors. *Machine Learning Journal*, 26(2):123–140, 1996.

[18] Yaov Freund and Robert E. Schapire. A decision-theoretic generalization of on-line and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1995.

[19] L. Breiman. Random forests. *Machine Learning Journal*, 45:5–32, 2001.

[20] T.K. Ho. The random subspace method for constructing decision forests. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 20:832–844, 1998.

[21] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 28:337–407, 2000.

[22] A. Fred. Finding consistent clusters in data partitions. In Josef Kittler and Fabio Roli, editors, *Multiple Classifier Systems, 3rd International Workshop on Muliple Classifier Systems MCS 2001, LNCS 2096*, pages 309–318. Springer, 2001.

[23] A. Fred and A.K. Jain. Data clustering using evidence accumulation. In *Proceedings of the 16th International Conference on Pattern Recognition. ICPR 2002*, volume 4, pages 276–280, Quebec City, Quebec, Canada, August 2002.

[24] A. Fred and A.K. Jain. Evidence accumulation clustering based on the k-means algorithm. In T.Caelli, A. Amin, R. Duin, M. Kamel, and D. de Ridder, editors, *Structural, Syntactic, and Statistical Pattern Recognition, volume LNCS 2396*, pages 442–451. Springer-Verlag, 2002.

[25] 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.

[26] H. G. Ayad and M. S. Kamel. Finding natural clusters using multi-clusterer combiner based on shared nearest neighbors. In *Multiple Classifier Systems: Fourth International Workshop, MCS 2003, UK, Proceedings.*, pages 166–175, 2003.

[27] H. G. Ayad, O. Basir, and M. S. Kamel. A probabilistic model using information theoretic measures for cluster ensembles. In *Multiple Classifier Systems: Fifth International Workshop, MCS 2004, Cagliari, Italy, Proceedings.*, pages 144–153, 2004.

[28] B. Fischer and J.M. Buhmann. Bagging for path-based clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1411–1415, 2003.

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

[30] A. Topchy, A.K. Jain, and W. Punch. Combining multiple weak clusterings. In *IEEE Intl. Conf. on Data Mining 2003, Proceedings*, pages 331–338, Melbourne, Fl., November 2003.

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

[32] S. Merugu and J. Ghosh. Privacy-preserving distributed clustering using generative models. In *IEEE Int'l Conf. on Data Mining (ICDM03)*, pages 211–218, Melbourne, FL, Nov 2003. AAAI/MIT Press.

[33] Evgenia Dimitriadou, Andreas Weingessel, and Kurt Hornik. A combination scheme for fuzzy clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 16(7):901–912, 2002.

[34] Kurt Hornik. A clue for cluster ensembles. Technical report, Department of Statistics and Mathematics Wirtschaftsuniversitt Wien, May 2005.

[35] X. Z. Fern and C. E. Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *20th International Conference on Machine Learning, Proceedings*, pages 186–193, Washington, DC., 2003.

[36] L. I. Kuncheva and S.T. Hadjitodorov. Using diversity in cluster ensembles. In *IEEE International Conference on Systems, Man and Cybernetics, Proceedings*, pages 1214–1219, The Hague, The Netherlands., 2004.

[37] T Lange and J. M. Buhmann. Combining partitions by probabilistic label aggregation. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 147–156. ACM, 2005.

[38] A. Topchy, A.K. Jain, and W. Punch. Clustering ensembles: Models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1866–1881, 2005.

[39] H. G. Ayad and M. S. Kamel. Cumulative voting consensus method for partitions with a variable number of clusters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):160–173, January 2008.

[40] 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. *Machine Learning Journal*, 52(1-2):91–118, 2003.

[41] A. Topchy, A.K. Jain, and W. Punch. A mixture model of clustering ensembles. In *SIAM Conf. on Data Mining*, pages 379–390, April 2004.

[42] A. D. Gordon and M. Vichi. Fuzzy partition models for fitting a set of partitions. *Psychometrika*, 66(2):229–248, 2001.

[43] A. Topchy, M. Law, A.K. Jain, and A. Fred. Analysis of consensus partition in clustering ensemble. In *IEEE Intl. Conf. on Data Mining 2004, Proceedings*, pages 225–232, Brighton, UK, 2004.

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

[45] B. Leclerc. Efficient and binary consensus functions on transitively valued relations. *Mathematical Social Sciences*, 8:45–61, 1984.

[46] D. A. Neumann and V. Norton. Clustering and isolation in the consensus problem for partitions. *Journal of Classification*, 3(2):281–297, 1986.

[47] B. Monjardet. Arrowian characterization of latticial federation consensus functions. *Mathematical Social Sciences*, 20:51–71, 1990.

[48] S. Régnier. Sur quelques aspect mathématique des problèmes de classification automatique, *ICC Bull.* 4:175-191, 1965, repr. *Mathématiques et Sciences Humaines*, 82:13–29, 1983.

[49] P. Arabie and S. A. Boorman. Multidimensional scaling of measures of distances between partitions. *Journal of Math. Psychol.*, 17:31–63, 1973.

[50] W.H.E Day. The complexity of computing metric distances between partitions. *Math. Soc. Sci.*, 1:269–287, 1981.

[51] J. P. Barthélemy and B. Monjardet. The median procedure in cluster analysis and social choice theory. *Mathematical Social Sciences*, 1:235–268, 1981.

[52] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2000.

[53] Evgenia Dimitriadou, Andreas Weingessel, and Kurt Hornik. Voting-merging: An ensemble method for clustering. In Georg Dorffner, Horst Bischof, and Kurt Hornik, editors, *Artificial Neural Networks-ICANN 2001*, pages 217–224, Vienna, Austria, August 2001. Springer.

[54] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, 1990.

[55] George Karypis and Vipin Kumar. Multilevel algorithms for multi-constraint graph partitioning. In *Conference on High Performance Networking and Computing. Proceedings of the 1998 ACM/IEEE conference on Supercomputing*, San Jose, CA, 1998.

[56] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. Technical Report TR 95-035, Department of Computer Science and Engineering, University of Minnesota, 1995.

[57] H. G. Ayad and M. S. Kamel. Refined shared nearest neighbors graph for combining multiple data clusterings. In *The 5th International Symposium on Intelligent Data Analysis IDA 2003. Berlin, Germany, Proceedings. LNCS. Springer.*, 2003.

[58] R.A. Jarvis and E.A. Patrick. Clustering using a similarity measure based on shared nearest neighbors. *IEEE Transactions on Computers*, C-22(11):1025–1034, November 1973.

[59] L. Ertoz, M. Steinbach, and V. Kumar. A new shared nearest neighbor clustering algorithm and its applications. In *Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining*, Arlington, VA, 2002.

[60] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood for incomplete data via the em algorithm (with discussion). *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[61] G. J. McLachlan and Krishnan T. *The EM Algorithm and Extensions*. Wiley, 1997.

[62] Christos H. Papadimitriou, Hisao Tamaki, Prabhakar Raghavan, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. In *ACM Conference on Principles of Database Systems (PODS)*, pages 159–168, Seattle, 1998.

[63] Samuel Kaski. Dimensionality reduction by random mapping: Fast similarity computation for clustering. In *Proceedings of IJCNN'98, International Joint Conference on Neural Networks*, volume 1, pages 413–418. IEEE Service Center, Piscataway, NJ, 1998.

[64] Dimitris Achlioptas. Database-friendly random projections. In *Symposium on Principles of Database Systems*, 2001.

[65] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Knowledge Discovery and Data Mining*, pages 245–250, 2001.

[66] M.A. Gluck and J.E. Corter. Information, uncertainty, and the utility of categories. In *Proceedings of the Seventh Annual Conference of the Cognitive Science Society*, pages 283–287, 1985.

[67] B. Mirkin. Reinterpreting the category utility function. *Machine Learning*, 45(2):219–228, 2001.

[68] H. G. Ayad and M. S. Kamel. On voting-based consensus of partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Under Review.

[69] Sanjoy Dasgupta. Experiments with random projection. In *UAI '00: Proceedings of the 16th Conference in Uncertainty in Artificial Intelligence*, pages 143–151, Stanford University, Stanford, California, USA, 2000.

[70] F. Leisch. Bagged clustering, 1999.

[71] B. Minaei, A. Topchy, and W. Punch. Ensembles of partitions via data resampling. In *IEEE Intl. Conf. on Information Technology: Coding and Computing, ITCC04, Proceedings*, volume 2, pages 188–192, Las Vegas, April 2004.

[72] H. G. Ayad and M. S. Kamel. Cluster-based cumulative ensembles. In *Multiple Classifier Systems: Sixth International Workshop, MCS 2005, Seaside, CA, USA, Proceedings.*, pages 236–245, 2005.

[73] L. I. Kuncheva and D. P. Vetrov. Evaluation of stability of k-means cluster ensembles with respect to random initialization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1798–1808, November 2006.

[74] Stefan Todorov Hadjitodorov, Ludmila I. Kuncheva, and Ludmila P. Todorova. Moderate diversity for better cluster ensembles. *Information Fusion*, 7(3):264–275, 2006.

[75] J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49(3):803821, 1993.

[76] C. Fraley and A.E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. Technical Report 329, Department of Statistics, University of Washington, 1995.

[77] J.C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3:32–57, 1973.

[78] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum, New York, 1981.

[79] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, USA, 1991.

[80] Fernando Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of English words. In *31st Annual Meeting of the ACL*, pages 183–190, 1993.

[81] Noam Slonim and Naftali Tishby. Agglomerative information bottleneck. In *NIPS*, pages 617–623, 1999.

[82] Inderjit S. Dhillon, Subramanyam Mallela, and Rahul Kumar. A divisive information-theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3:1265–1287, 2003.

[83] J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1995.

[84] William H. Wolberg and O.L. Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. In *Proceedings of the National Academy of Sciences*, volume 87, pages 9193–9196, U.S.A., December 1990.

[85] P. Baldi and G. W. Hatfield. *DNA Microarrays and Gene Expression. From Experiments to Data Analysis and Modelling*. Cambridge University Press, 2002.