Semi-supervised Information Fusion for Clustering, Classification and Detection Applications

SEMI-SUPERVISED INFORMATION FUSION FOR CLUSTERING, CLASSIFICATION AND DETECTION APPLICATIONS

BY

HUAYING LI, M.A.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Doctor of Philosophy (2017) McMaster University
(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE: Semi-supervised Information Fusion for Clustering, Classification and Detection Applications

AUTHOR: Huaying Li
M.A.Sc., (Electrical & Computer Engineering)
McMaster University, Hamilton, Canada

SUPERVISOR: Dr. Aleksandar Jeremic

NUMBER OF PAGES: xiv, 149

Abstract

Information fusion techniques have been widely applied in many applications including clustering, classification, detection and etc. The major objective is to improve the performance using information derived from multiple sources as compared to using information obtained from any of the sources individually. In our previous work [1], we demonstrated the performance improvement of Electroencephalography(EEG) based seizure detection using information fusion. In the detection problem, the optimal fusion rule is usually derived under the assumption that local decisions are conditionally independent given the hypotheses. However, due to the fact that local detectors observe the same phenomenon, it is highly possible that local decisions are correlated. To address the issue of correlation, we implement the fusion rule sub-optimally by first estimating the unknown parameters under one of the hypotheses and then using them as known parameters to estimate the rest of unknown parameters.

In the aforementioned scenario, the hypotheses are uniquely defined, i.e., all local detectors follow the same labeling convention. However, in certain applications, the regions of interest (decisions, hypotheses, clusters and etc.) are not unique, i.e., may vary locally (from sources to sources). In this case, information fusion becomes more complicated. Historically, this problem was first observed in classification and clustering. In classification applications, the category information is pre-defined and training data is required. Therefore, a classification problem can be viewed as a detection problem by considering the pre-defined classes as the hypotheses in detection. However, information fusion in clustering applications is more difficult due to the lack of prior information and the correspondence problem caused by symbolic cluster labels.

In the literature, information fusion in clustering problem is usually referred to as clustering ensemble problem. Most of the existing clustering ensemble methods are unsupervised. In this thesis, we proposed two semi-supervised clustering ensemble algorithms (SEA). Similar to existing ensemble methods, SEA consists of two major steps: the generation and fusion of base clusterings. Analogous to distributed detection, we propose a distributed clustering system which consists of a base clustering generator and a decision fusion center. The role of the base clustering generator is to generate multiple base clusterings for the given data set. The role of the decision fusion center is to combine all base clusterings into a single consensus clustering. Although training data is not required by conventional clustering algorithms (usually unsupervised), in many applications expert opinions are always available to label a small portion of data observations. These labels can be utilized as the guidance information in the fusion process. Therefore, we design two operational modes for the fusion center according to the absence or presence of the training data. In the unsupervised mode, any existing unsupervised clustering ensemble methods can be implemented as the fusion rule. In the semi-supervised mode, the proposed semi-supervised clustering ensemble methods can be implemented. In addition, a parallel distributed clustering system is also proposed to reduce the computational times of clustering high-volume data sets. Moreover, we also propose a new cluster detection algorithm based on SEA. It is implemented in the system to provide feedback information. When data observations from a new class (other than existing training classes) are detected, signal is sent out to request new training data or switching from the semi-supervised mode to the unsupervised mode.

Acknowledgments

I wish to express my deep and sincere gratitude to my supervisor, Dr. Aleksandar Jeremic, for his encouragement and guidance from the initial to the final stage of my graduate study and for his support and detailed comments on the completion of this thesis.

I wish to sincerely thank all my supervisory committee members, Dr. Jian-Kang Zhang and Dr. Nicola Nicolici, for their valuable comments and suggestions. Special thanks go to Dr. David Andrews, University of Toronto, for providing me real data set to conduct the experiments. I also wish to thank all the staff members in the ECE department, especially Cheryl Gies.

Last, but far from least, I express my deep appreciation to my parents and my husband for their continuous love and support. I also would like to thank my daughters who bring immeasurable joy and happiness into my life.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Information fusion is an active research topic whose aim is to improve performance (commonly detection and/or prediction) by combining information/data obtained from multiple sources. The sources include, but are not limited to, physical sensors (such as radar and infrared) or other devices (such as cameras and cell phones). In addition, humans (such as expert opinions and intelligence gathering) and data archives (such as social media posts and website documents) can also be viewed as sources. Information fusion techniques have been widely applied in a variety of applications such as business intelligence, financial forecasting, security surveillance and distributed detection [2, 3, 4, 5, 6]. As stated before, one of the most important goals of fusion is to achieve improved results (decisions, actions, labels and etc.) using information derived from multiple sources as compared to using information obtained from any individual source. For example, in economic forecasting, economists combine multiple forecasts to produce a superior forecast [7]. In automatic recognition, a multi-modal biometric system consolidates the evidence provided by multiple biometric sources (fingerprint, retina, face, voiceprint and etc.) in order to improve the

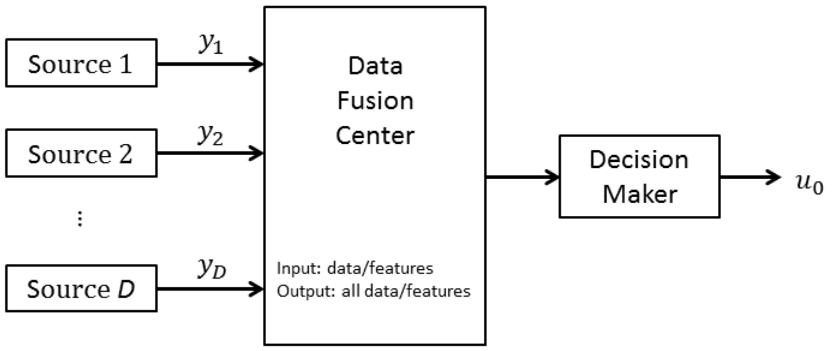


Figure 1.1: Centralized information fusion system

recognition performance as compared to a single biometric modality [8].

The fusion process can be classified into two major groups according to the distribution of information: (a) centralized and (b) decentralized. In a centralized system, as shown in Figure 1.1, information derived from each source is sent to a central unit. The central unit is usually referred to as the data fusion center and receives all data transmitted from multiple sources. The final decision is then made by the decision maker using all of the available information. In a decentralized system, as shown in Figure 1.2, data is pre-processed by local decision makers before being transmitted

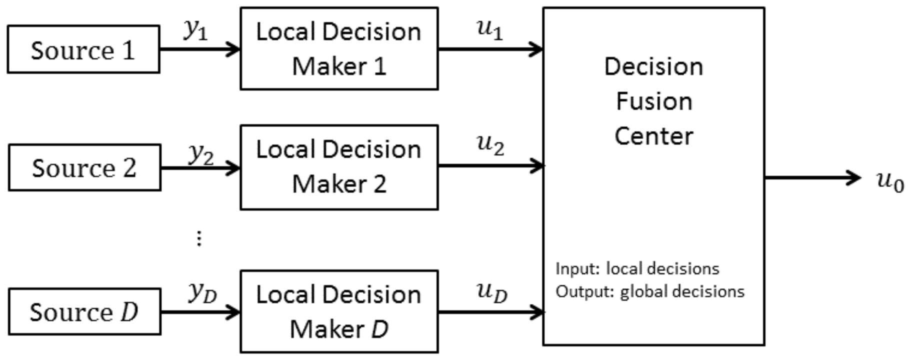


Figure 1.2: Decentralized information fusion system

to a central unit. In such a system, the central unit is often referred to as the decision fusion center. The final decision is made by the decision fusion center using a condensed version of the information.

Theoretically speaking, centralized systems perform better in terms of accuracy since the decision maker has the access to all information. However, the requirement on the system computational capability increases dramatically when the amount of information becomes large, because the decision maker uses conventional statistical techniques to make decisions based on all information. Moreover, the applicability of a centralized system is always restricted by communication bandwidth and data transmission speed. As a consequence, decentralized systems become more attractive and preferable since information is transmitted in a condensed format (e.g., local decisions) and the requirement on communication bandwidth and transmission speed is reduced. In addition, the computational complexity is reduced by distributing the computational load of the central unit over the local decision makers. Although the performance of decentralized system is reduced due to the lack of some information, the performance loss is negligible and could be minimized by applying optimal decision rule in the local decision makers and optimal fusion rule in the fusion center [9, 10].

The fusion objectives usually vary among the applications and include but are not limited to: detection of an event, tracking of an object, classification of observations, clustering of a data set and identification of key words from a text document. In this thesis, we focus on information fusion techniques used in detection, classification and clustering applications.

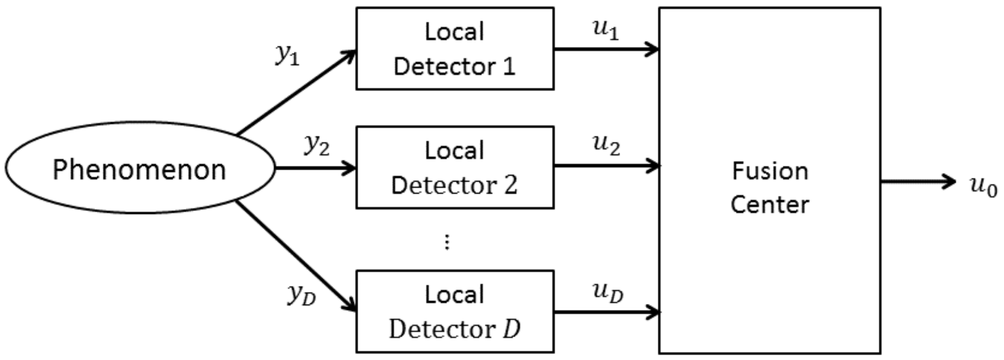


Figure 1.3: Distributed detection system with multiple local detectors

1.1 Distributed Detection

Information fusion techniques can be applied in a distributed detection system with multiple sensors to improve the accuracy of detection. A typical distributed detection system consists of multiple local detectors and a fusion center, as shown in Figure 1.3. The local detectors observe the same phenomenon and make local decisions based on their own observations. Local decisions are sent to the fusion center for further processing. In many detection problems, decision making can be formulated as selecting a choice between several possibilities, which are often referred to as hypotheses. The optimal fusion rule based on minimizing the overall probability of error was proposed in [11] for the simplest case of two hypotheses H_0 and H_1. The common assumption employed in distributed detection is that the local observations are conditionally independent in the sense that $P(y_i, y_j|H_k) = P(y_i|H_k)P(y_j|H_k)$ for all $i \neq j$ and all $k = 0$ or 1. When local decision rules are given, the optimal fusion rule [11] shows that optimal global decision is determined by the prior probabilities of hypotheses and the probabilities of false alarm and missed detection of local sensors. Unfortunately, in real applications, these probabilities are usually unknown or may be time-varying. To implement the optimal fusion rule, algorithms on estimating these unknown probabilities have been proposed in the literature. The estimator in [12] is asymptotically biased and requires biased-correction to reduce the estimation error. It increases the computational complexity of the algorithm. The blind algorithm proposed in [13] estimates the unknown probabilities by analytically solving a set of nonlinear equations consisting of the probabilities of different local decision combinations. Although in real applications, these probabilities of decision combinations are not available, they can be estimated by time averaging, i.e., the empirical probabilities. This blind adaptive estimator [13] is unbiased, more reliable, and more efficient compared with the one proposed in [12]. On the other hand, the optimal fusion rule [11] shows that global decision is made by comparing the weighted sum of local decisions with a threshold. Each weight is a function of the probabilities of false alarm and missed detection. The threshold is determined by the probabilities of the hypotheses. The algorithms proposed in [14] and [15] estimate the weights directly instead of estimating the unknown probabilities. In order to reduce the estimation error, fused decisions of all other local sensors are classified either as reliable or unreliable. Consequently, unreliable global decisions are not used to update the weights. However, dropping such decisions may deteriorate the algorithm's convergence behavior. The lack of a proper procedure of selecting the reliable range may limit the applicability of these algorithms.

The more general detection problem with multiple hypotheses is often referred to as the M-ary distributed detection problem. The extended version of the aforementioned optimal fusion rule [11] was proposed in [16] by assuming the probabilities of anomalies conditioned on different hypotheses to be the same for a particular detector. The more general optimal fusion rule was proposed in [17] by defining $(M-1)$ different probabilities of anomalies conditioned on each hypothesis. To implement the optimal fusion rule, the unknown prior probabilities of hypotheses and the probabilities of anomalies of local sensors are estimated using the nonlinear least square estimator and the maximum likelihood estimator. The authors showed that the maximum likelihood estimator performs more accurately than the least square estimator when the number of available decisions is small. Another approach [18] to implement the optimal fusion rule is to convert the M-ary problem into a sequence of binary problems. The basic idea is to sequentially reduce the M-ary object space in multiple stages until it contains exactly one object. At each stage, two objects are tested against each other and one of them is rejected.

1.2 Distributed Clustering

Information fusion techniques can be applied in the classification and clustering applications as well. Classification is the task of identifying the category membership for a new observation, when a set of data observations is available for training and their category membership is known. It can be viewed as a supervised learning and its goal is to formulate a mapping function between a training set and its known labels. This mapping function can be used for mapping new data. Although a variety of algorithms are usually available to solve a classification problem, the comparative accuracy of these algorithms highly depends on the problem that needs to be solved. Due to the fact that statistical properties of a given data set are usually unavailable, it is difficult to select a suitable algorithm for a particular problem. Therefore, the combination of multiple classifiers is an option to improve the accuracy of a single classifier, shown in studies in [19, 20, 21]. The classification problem with M pre-defined categories could be viewed as the M-ary hypothesis testing problem in which the k-th hypothesis is true if data point $\mathbf{x}$ belongs to the k-th cluster where $k = 1, ..., M$. Therefore, the problem involving the combination of multiple classifiers can be treated as the distributed detection problem.

In contrast to classification, clustering can be viewed as a typical example of unsupervised learning. Usually in clustering problems no categories or labels are predefined. The goal of clustering is to find the hidden structure of a given data set by dividing data points into distinct clusters based on certain criteria. Data points in the same cluster are expected to be more similar to each other than to a data point from another cluster. Although many clustering algorithms exist in the literature, in practice no single algorithm can correctly identify the underlying structure of all data sets [22, 23]. Furthermore, it is usually difficult to select a suitable clustering algorithm for a given data set when the prior information about cluster shape and size is not available. In addition, a particular clustering algorithm usually generates different clusterings for a given data set by starting from different initiations or using different parameter settings. Consequently, we expect to improve the quality of cluster analysis by combining multiple clusterings into a consensus clustering. The problem involving the combination of multiple clusterings is often referred to as clustering ensemble problem in the literature [24, 25, 26, 27, 28].

Commonly clustering ensemble methods consist of two major steps: generation and fusion of multiple base clusterings. In the generation step, there are different ways to generate a set of clusterings, such as applying different clustering algorithms, applying the same clustering algorithm with different initiations, and applying the clustering algorithm(s) to different combinations of features. In the fusion step, multiple base clusterings are combined into a single consolidated clustering by a particular fusion rule. Different from the distributed detection problem, information fusion in cluster analysis is more difficult because of at least the following two reasons: (1) the number of clusters in each clustering could be different and the desired number of clusters is usually unknown; and (2) the cluster labels from different clusterings are symbolic and the same symbolic label from different clusterings sometimes corresponds to different clusters. Therefore, a correspondence problem is always accompanied with clustering ensemble problem [24]. The common way to avoid the correspondence problem [29, 30] is to construct a pairwise similarity matrix between data points. The i, j-th entry of the similarity matrix is 1 if data points $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same cluster and 0 if they belong to different clusters. Each local clustering corresponds to one similarity matrix. The accumulative evidence of similarity is calculated by averaging all of the matrices and used as the input of a final clustering algorithm. In [24], the authors proposed three algorithms based on the hypergraph representation of clusterings to solve the ensemble problem. These algorithms are named as cluster-based similarity partitioning algorithm (CSPA), hypergraph partitioning algorithm (HGPA) and meta-clustering algorithm (MCLA), respectively. In MCLA, the clusters of a local clustering are represented by hyperedges. Hyperedges from all local clusterings are grouped into meta-clusters, each of which contains similar hyperedges. Data points are assigned to the collapsed hyperedge (cluster) they most strongly belong to. Many other approaches to combine the base clusterings have been proposed in the literature, such as relabeling and voting based and mixture-densities based approach [29, 31, 32, 26].

1.3 Contributions

The major objective of this thesis is to improve the performance (detection and classification) using the proposed distributed systems and information fusion techniques. The specific contributions of this thesis are stated as follows:

- Distributed Detection

  - In the distributed detection system, we implement the optimal fusion rule formulated for correlated local decisions [33] in the fusion center, where global decisions are determined by the prior probabilities of hypotheses, the probabilities of false alarm and missed detection of local sensors, and additional correlation coefficients. We propose to estimate these unknown parameters in a sub-optimal way by first estimating the correlation coefficients under H_0 (in the absence of seizures) and then utilizing the estimates as known parameters when seizures are present. In addition when seizure is absent, we can also estimate the probabilities of false-alarm of local detectors and thus decrease the number of unknown parameters further.

- Distributed Clustering

  - We propose semi-supervised clustering ensemble algorithms: soft-to-hard semi-supervised clustering ensemble algorithm (SHSEA) and hard-to-hard semi-supervised clustering ensemble algorithm (HHSEA). The former is to produce hard consensus clustering based on computing the associations between data points and reference clusters (associations are represented using a soft label matrix). The latter is to produce hard consensus clustering based on relabeling the set of base clusterings according to the reference labels (relabeled base clusterings are represented in a hard label matrix).
  - A distributed clustering system is proposed, which consists of a base clustering generator and a fusion center. We design two operational modes for the fusion center: unsupervised and semi-supervised mode, according to the absence and presence of labeled training data.

- We propose four base clustering generators by changing the input of local clusterers and/or varying the number of clusters generated in each base clustering. The effect of base clusterings on clustering ensemble is illustrated by an empirical evaluation of the proposed distribution clustering system.
- A semi-supervised feature selection algorithm is proposed based on estimating and ranking the performance of local clusterers using reference data observations and their labels.
- To clustering high-volume data sets we propose a parallel distributed clustering system, which contains multiple distributed clustering systems. The input data set is divided into smaller subsets and all subsets are processed in parallel. The advantages of such a system is that it retains the performance of distributed clustering system and meanwhile it reduces the computational time for clustering high-volume data sets.
- A new cluster detection algorithm is proposed to detect the occurrence of data observations from a new class other than existing training classes. This algorithm is implemented to provide the feedback for the system. When data observations from a new class is detected, signal is sent out to request additional training data (of new class) or switching from semi-supervised mode to unsupervised mode.
- We demonstrate the performance of our proposed systems and the effect of data pre-processing techniques to the clustering and clustering ensemble methods.

1.4 Outline

In the following chapter, the distributed detection system designed for EEG seizure detection problem is discussed. In Chapter 3 we review several existing clustering and clustering ensemble algorithms and propose the semi-supervised clustering ensemble methods. The distributed clustering system is proposed in Chapter 4. New cluster detection algorithm and the parallel distributed clustering system are also proposed in this chapter. In Chapter 5, we provide numerical examples to evaluate the distributed detection system and the distributed clustering system. The final chapter concludes this thesis with a summary of the contributions to cluster analysis and presents some future research directions.

Chapter 2

Distributed Detection System

Data fusion techniques can be used in distributed detection to improve the performance and reduce the detection error. In our previous work, we utilized the distributed detection system to detect seizure activities in neonatal EEG [34]. A seizure is defined clinically as a paroxysmal alteration in neurological function, i.e., behavioral, motor, or autonomic function. It is a result of excessive electrical discharges of neurons, which usually develop synchronously and happen suddenly in the central nervous system (CNS). It is critical to recognize seizures in newborns, since they are usually related to other significant illnesses. Seizures are also an initial sign of neurological disease and a potential cause of brain injury [35].

At the beginning of this chapter, we briefly describe the structure of the distributed detection system. Next, we introduce the decision rules applied in local detectors. In Section 2.3, we discuss the optimal fusion rules that can be implemented in the fusion center. To further improve the performance of the distributed detection system, we remove the standard assumption employed in distributed detection and implement the optimal fusion rule proposed in [33], which is designed for correlated local decisions.

2.1 Overview of the System

The phenomenon, multiple sensors and their local processing units and a fusion center are the basic components of a typical distributed detection system. In our particular application, neonatal EEG signals make up the phenomenon and the local processing units are called local detectors. Three local detectors are employed in the system as shown in Figure 2.1. Additional detectors can be added into the system whenever more information is required to make the final decision. Although increasing the number of detectors has the potential to reduce the overall probability of error, it may also increase the complexity of computation.

The role of local detectors is to observe the EEG signals and determine if there are seizure activities in the EEG signal or not based on their own decision rules. The local decisions of each detector are sent to the fusion center. The role of the fusion center is to make the final decisions based on all local decisions using a certain fusion rule. The fusion center has no access to the original EEG signal.

2.2 Local Detectors

A variety of seizure detection algorithms exist in the literature [36]. We implemented and applied three popular algorithms [37, 38, 39] as the local decision rules. The basic ideas of these algorithms are listed as follow:

- **Liu's algorithm** - In [37], the authors focused on the rhythmic characteristic of neonatal EEG seizure and proposed a detection algorithm using autocorrelation analysis. Due to the periodicity of EEG seizure, its autocorrelation function has more peaks with similar periodicity of the original signal. In contrast, normal

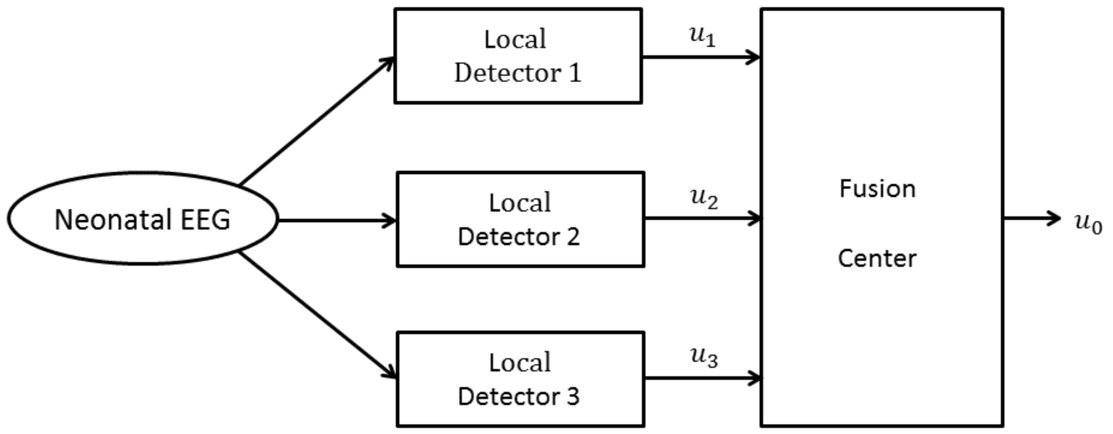


Figure 2.1: Neonatal EEG seizure detection system

neonatal EEG does not have clear periodicity, so its autocorrelation usually has irregular peaks. A scoring system described in can be used to determine the degree of periodicity of the EEG signal quantitatively in order to identify the existences of the seizure activities.

- **Gotmans's algorithm** - In [38], the authors proposed three different seizure detection methods to detect three types of seizures: rhythmic discharges, multiple spikes, and very slow rhythmic discharges, respectively. In this thesis, we only focus on the rhythmic discharge detection since it could identify 90% of the seizures detected by all three detection algorithms. The rhythmicity of a signal can be represented in the frequency domain by a high and narrow peak at the frequency of that signal. Therefore, in the spectrum of the EEG segment containing seizure activities, a large distinct peak is expected to appear at the main frequency of EEG seizure.

- **Celka's algorithm** - In [39], the authors performed the singular spectrum analysis and the information theoretic-based signal subspace selection to examine the complexity of the EEG signal. This detection algorithm has three main steps: pre-processing, singular spectrum analysis, and minimum description length.

We use these algorithms to formulate the local decision rules and perform hypothesis testing with two hypotheses:

$$\begin{aligned} H_0: &\quad \text{The EEG signal does not contain seizure} \\ H_1: &\quad \text{The EEG signal contains seizure.} \end{aligned}$$

For the j-th local detector LD_j, $j = 1, 2, 3$, local decision $u_j = 0$ if it favors the hypothesis H_0 and $u_j = 1$ if it favors the hypothesis H_1. A common assumption made for distributed detection problem is that the local observations y_j are conditionally independent in the sense that $P(y_i, y_j|H_k) = P(y_i|H_k)P(y_j|H_k)$ for all $i \neq j$ and all k.

2.3 Fusion Center

Under the assumption that local detectors are conditionally independent given the hypotheses, the optimal fusion rules have been well documented in the literature [11, 16, 33]. In this section, we review the optimal fusion rules based on binary hypothesis and M-ary hypothesis testing respectively [11, 16], which were previously implemented in the system as the optimal fusion rule. Since the local detectors observe the same phenomenon and exploit similar properties of the signal, it is possible that local decisions are statistically dependent or correlated. Therefore, in this thesis we implement the more general fusion rule proposed in [33] to further improve the performance of the distributed detection system by removing the standard assumption employed in distributed detection.

2.3.1 Binary Hypothesis Testing

For a binary hypothesis testing problem, the overall probability of error of the system P_e is calculated by

$$P_e = P(H_0) \cdot P(u_0 = 1|H_0) + P(H_1) \cdot P(u_0 = 0|H_1), \tag{2.1}$$

where $P(H_0)$ and $P(H_1)$ are the a priori probability of hypothesis H_0 and H_1, respectively. The sum of $P(H_0)$ and $P(H_1)$ equals to one. The global decision is $u_0 = 0$ if the system favors H_0. The global decision is $u_0 = 1$ if it favors the hypothesis H_1. For a system containing D local detectors, the authors of [11] provide the optimal fusion rule, Eq. (2.2)-(2.4), based on minimizing the overall probability of error.

$$\begin{aligned} u_0 &= \begin{cases} 1, & \text{if } a_0 + \sum_{j=1}^{D} a_j > 0 \\ 0, & \text{otherwise} \end{cases} && (2.2) \\ \text{where,} \quad a_0 &= \log \frac{P(H_1)}{P(H_0)} && (2.3) \\ \text{and} \quad a_j &= \begin{cases} \log \frac{1-P_j^m}{P_j^f}, & \text{if} \quad u_j = 1 \\ \log \frac{P_j^m}{1-P_j^f}, & \text{if} \quad u_j = 0 \end{cases} && (2.4) \end{aligned}$$

Here P_j^f and P_j^m represent the probabilities of false alarm and missed detection of the j-th detector respectively, i.e.,

$$P_j^f = P(u_j = 1 | H_0) \tag{2.5}$$

$$P_j^m = P(u_j = 0 | H_1) \tag{2.6}$$

The optimal fusion rule shows that the global decisions u_0 is determined by local decisions u_j, the a priori probability $P(H_0)$ or $P(H_1)$, and the probabilities of false alarm P_j^f and miss detection P_j^m of local detectors. These probabilities are unknown in our seizure detection problem. They are usually unavailable in many other real applications. In order to make the final decisions, we would like to utilize the information available to us, the local decisions, to estimate these unknown parameters.

To implement the optimal rule, the blind adaptive algorithm proposed in [13] estimates the unknown probabilities by analytically solve a set of nonlinear equations consisting of the probabilities of different local decision combinations. In our distributed system with three local detectors (i.e. $D = 3$), the local decision combination $\mathbf{u} = \{u_1 = s_1,\ u_2 = s_2,\ u_3 = s_3\}$ has $L = 2^3$ possible outcomes where $s_1, s_2, s_3 = 0$ or 1. The probability of the ℓ-th combination, $\ell = 1, \ldots, L$, is denoted as P_ℓ and calculated by

$$\begin{aligned} P_\ell &= \Pr(u_1 = s_1, u_2 = s_2, u_3 = s_3) \\ &= P(u_1 = s_1|H_0)P(u_2 = s_2|H_0)P(u_3 = s_3|H_0)P(H_0) \\ &\quad + P(u_1 = s_1|H_1)P(u_2 = s_2|H_1)P(u_3 = s_3|H_1)P(H_1), \end{aligned} \tag{2.7}$$

where

$$P(u_j = s_j|H_0) = \begin{cases} P_j^f, & \text{if} \quad s_j = 1 \\ 1 - P_j^f, & \text{if} \quad s_j = 0 \end{cases} \tag{2.8}$$

$$P(u_j = s_j|H_1) = \begin{cases} 1 - P_j^m, & \text{if} \quad s_j = 1 \\ P_j^m, & \text{if} \quad s_j = 0 \end{cases} \tag{2.9}$$

By substituting Eq. (2.8), Eq. (2.9) and $P(H_0) = 1 - P(H_1)$ into Eq. (2.7), it generates a nonlinear system of eight equations with seven unknowns, $P(H_1)$, P_j^m and P_j^f for $j = 1, 2, 3$. Seven out of these eight equations are independent because of $\sum P_\ell = 1$. The nonlinear system is solvable when P_ℓ are known. Although P_ℓ is usually unavailable in practice, it could be replaced by empirical probability defined as

$$\begin{aligned} \hat{P}_\ell &= \Pr(u_1 = s_1, u_2 = s_2, u_3 = s_3) \\ &\simeq \frac{\text{number of}\,\{u_1 = s_1, u_2 = s_2, u_3 = s_3\}}{N} \end{aligned} \tag{2.10}$$

where N is the total number of decisions made by one of the local detectors. The analytical solution to the above nonlinear equations is given in [13]. However, the usage of empirical probabilities (Eq. (2.10)) is limited when the number of decisions is not large enough. In our particular seizure detection problem, the number of seizures occurring can be rather small. Therefore, empirical probabilities in Eq. (2.10) may yield inaccurate estimation results. Therefore, we illustrate another way of estimating the unknown parameters in the next section.

2.3.2 M-ary Hypothesis Testing

To estimate the unknown probabilities required for implementing the optimal fusion rule, a maximum likelihood estimator has been proposed in [17]. We review this algorithm by considering an M-ary hypothesis testing problem and apply it in our seizure detection problem with $M = 2$.

In a more general problem with more choices of decisions, the M-ary hypothesis testing is usually involved and the hypotheses are denoted by $H_0, H_1, \ldots, H_{M-1}$. The *a priori* probability of hypothesis H_k is denoted by $P(H_k)$, where $k = 0, \ldots, M-1$. Here we adopt the convention $u_j = k$ if the j-th detector favors H_k. Similarly, $u_0 = k$

if the system favors H_k. The overall probability of error of the system P_e defined in Eq. (2.1) becomes

$$P_e = \sum_{i=0}^{M-1} \sum_{\substack{k=0 \\ k \neq i}}^{M-1} P(H_i) P(u_0 = k | H_i). \tag{2.11}$$

In [17], the probability of anomaly of the j-th local detector is used to measure its performance and defined by

$$\varepsilon_{ik}^{j} \triangleq P(u_j = k | H_i) \tag{2.12}$$

where $i, k \in \{0, \ldots, M-1\}$ and $i \neq k$. When $i = k$, the corresponding probability ε_{ii}^{j} becomes the probability of correctness of the j-th detector and is defined by

$$\varepsilon_{ii}^{j} \triangleq P(u_j = i | H_i) = 1 - \sum_{\substack{k=0 \\ k \neq i}}^{M-1} \varepsilon_{ik}^{j} \tag{2.13}$$

It has been shown in [40] that minimizing P_e in Eq. (2.11) is equivalent to maximizing the posterior probability

$$P(H_i | \mathbf{u}) = P(H_i | u_1, \ldots, u_D) = \frac{P(H_i)}{P(\mathbf{u})} P(u_1 | H_i) \cdots P(u_N | H_i), \tag{2.14}$$

where $\mathbf{u} = \{u_1, \ldots, u_D\}$. For $i = 0, \ldots, M-1$, the global decision u_0 is determined by

$$u_0 = \arg\max_{H_i} P(H_i | \mathbf{u}) = \arg\max_{H_i} P(H_i) \prod_{j \in S_0} \varepsilon_{i0}^{j} \cdots \prod_{j \in S_{M-1}} \varepsilon_{iM-1}^{j}, \tag{2.15}$$

where $S_0, \ldots, S_{M-1}$ represents the partition of the indices of the local detectors, i.e.,

$$\begin{aligned} S_0 &\triangleq \{j|u_j = 0, \forall j = 1, \ldots, N\} \\ &\cdots \\ S_{M-1} &\triangleq \{j|u_j = M-1, \forall j = 1, \ldots, N\} \end{aligned} \tag{2.16}$$

Eq. (2.15) shows that the optimal global decision is determined by the prior probabilities of the hypotheses and the probabilities of anomaly defined in Eq. (2.12). These probabilities are usually unknown in real applications. We need to estimate these unknown parameters using the local decisions.

In a distributed detection system with D local detectors, the local decision combination, $\mathbf{u} = \{u_1 = s_1, \ldots, u_j = s_j, \ldots, u_D = s_D\}$, has $L = M^D$ possible outcomes where $s_j \in \{0, \ldots, M-1\}$. Suppose the occurrence number of the ℓ-th decision combination is denoted by a random variable X_ℓ and the corresponding occurrence probability is denoted by P_ℓ, where $\ell = 1, \ldots, L$. Under the standard assumption employed in the distributed detection, the joint probabilities P_ℓ can be calculated using Bayes' rule and written by

$$\begin{aligned} P_\ell &= \Pr(u_1 = s_1, \ldots, u_j = s_j, \ldots, u_D = s_D) \\ &= \sum_{i=0}^{M-1} P(H_i) P(u_1 = s_1, \ldots, u_D = s_D | H_i) \\ &= \sum_{i=0}^{M-1} P(H_i) P(u_1 = s_1 | H_i) \cdots P(u_D = s_D | H_i). \end{aligned} \tag{2.17}$$

Therefore, Eq. (2.17) is a function of $M-1$ prior probabilities and $(M-1)MD$

probabilities of anomaly. All of these probabilities are unknown in many real applications. We define a vector $\boldsymbol{\theta}$ consisting all the unknown parameters. The dimension of vector $\boldsymbol{\theta}$ is $(M-1)(MD+1)$. Thus, Eq. (2.17) could be written in a short form, i.e., $P_\ell = f_\ell(\boldsymbol{\theta})$.

For a fixed total number of local decisions N, the occurrence numbers $\mathbf{X} = (X_1, \ldots, X_L)$ are multinomially distributed with probability mass function

$$P(X_1 = x_1, \ldots, X_L = x_L | N) = \frac{N!}{x_1! \cdots x_L!} P_1^{x_1} \cdots P_L^{x_L}, \tag{2.18}$$

where $\text{Var}(X_\ell) = NP_\ell(1-P_\ell)$ and $\text{Cov}(X_s, X_\ell) = -NP_\ell(1-P_\ell)$ for $s, \ell = 1, \ldots, L$ and $s \neq \ell$. Once the occurrence numbers are known, the maximum likelihood (ML) estimation is an efficient way to estimate the unknown parameters defined in vector $\boldsymbol{\theta}$. The estimate $\hat{\boldsymbol{\theta}}$ is given by

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} P(X_1 = x_1, \ldots, X_L = x_L | N, \boldsymbol{\theta}). \tag{2.19}$$

To solve this particular neonatal seizure detection problem, we use the ML estimator defined by Eq. (2.19) with $M = 2$ and $D = 3$, i.e., the dimension of vector $\boldsymbol{\theta}$ is seven and the number of possible decision combination, L, equals to eight.

2.3.3 Correlated Local Decisions

Since the existing local detectors exploit similar properties of the EEG signal it is very likely that the local decisions are statistically dependent. Consequently the overall performance of the detection system can be sub-optimal if this correlation is not properly accounted for. To this purpose we implement the optimal fusion rule proposed in [33] in the fusion center, which was developed for correlated local decisions.

Suppose the decision combination $\{u_1 = s_1, \ldots, u_D = s_D\}$ where $s_1, \ldots, s_D \in \{0, 1\}$ is denoted by a vector $\mathbf{u} = [u_1, \ldots, u_D]^T$ and its probability density function is given in [33] by

$$P(\mathbf{u}) = P_1(\mathbf{u}) \left[1 + \sum_{i<j} \gamma_{ij} z_i z_j + \sum_{i<j<k} \gamma_{ijk} z_i z_j z_k + \cdots + \sum \gamma_{12\ldots D} z_1 z_2 \ldots z_D \right] \quad (2.20)$$

where

$$\begin{aligned} P_1(\mathbf{u}) &= \prod_{i=1}^{D} {p_i}^{u_i} (1 - p_i)^{1-u_i} \\ p_i &= \Pr(u_i = 1). \end{aligned}$$

Also, z_i is a normalized random variable with zero mean and unit variance, i.e.,

$$z_i = \frac{u_i - p_i}{\sqrt{p_i(1 - p_i)}}$$

and γ_{ij}, γ_{ijk} and $\gamma_{12\ldots D}$ are the second, third and D-th order correlation coefficients respectively and defined as

$$\begin{aligned} \gamma_{ij} &= \mathrm{E}(z_i z_j) \\ \gamma_{ijk} &= \mathrm{E}(z_i z_j z_k) \\ \gamma_{12\ldots D} &= \mathrm{E}(z_1 z_2 \ldots z_D) \end{aligned}$$

In this particular neonatal EEG seizure detection problem, we observe that the decision vector $\mathbf{u}$ is a multivariate binomial vector. Its probability density function, Eq. (2.20), can be approximated as

$$P(\mathbf{u}) = P_1(\mathbf{u}) \left[1 + \sum_{i<j} \gamma_{ij} z_i z_j \right] \tag{2.21}$$

by neglecting the third and higher order correlation coefficients.

It has been shown in [33] that optimal fusion rule is given by

$$u_0 = \begin{cases} 1 & \log\lambda(\mathbf{u}) > \log \frac{P(H_0)}{P(H_1)} \\ 0 & \text{otherwise} \end{cases} \tag{2.22}$$

where

$$\begin{aligned} \log\lambda(\mathbf{u}) &= \sum_{i=1}^{D} u_i \left[\log \frac{(1-P_i^m)(1-P_i^f)}{P_i^m P_i^f} \right] \\ &+ \sum_{i=1}^{D} \log \frac{P_i^m}{(1-P_i^f)} + \log \frac{1+\sum_{i<j} \gamma_{ij}^1 z_i^1 z_j^1}{1+\sum_{i<j} \gamma_{ij}^0 z_i^0 z_j^0} \\ z_i^h &= \frac{u_i - P(u_i = 1|H_h)}{\sqrt{P(u_i = 1|H_h)(1 - P(u_i = 1|H_h))}} \quad h = 0,1 \\ \gamma_{ij}^h &= \sum_{\mathbf{u}} z_i^h z_j^h P(\mathbf{u}|H_h) \quad h = 0,1 \end{aligned}$$

Unlike the case of uncorrelated decisions, the optimal fusion rule expressed in Eq. (2.22) includes six additional unknown parameters (correlation coefficients). Each correlation coefficient is hypothesis dependent. To avoid large number of unknown parameters, we propose to estimate them under H_0 (in the absence of seizures) and then treat them as known parameters when seizures are present. In addition, during the period of absence of seizures we can estimate probabilities of false alarm. In this way, the number of unknown parameters is reduced even further.

The preliminary results [17] indicate that, when the anomalies do not change and the relative occurrence of seizures is sufficient, the performance can be improved by treating the problem in a semi-blind way. Note that by a semi-blind way, we refer to the mode of operation in which statistical anomalies in the absence of seizures are estimated beforehand. Once the seizures start to occur, these parameters are treated as known. Thus, in a correlated scenario we may be better off by treating the problem in the semi-blind way using estimates of statistical properties in the absence of seizures.

Chapter 3

Data Clustering

Grouping objects into meaningful categories is a fundamental mode of understanding and learning. Humans are excellent on performing this task. For example, we learn to organize and divide a basket of blocks into different shape groups in our early childhood. Although humans can easily identify clusters in two and/or three dimensions, we need automatic algorithms to partition the high-volume and high-dimensional data sets. Clustering algorithms are developed to perform this task in many fields, such as in pattern recognition, bioinformatics, data mining, image segmentation, and information retrieval [41, 42, 43, 44, 45].

Nowadays in the Big Data era high-volume and high-dimensional data sets are being created continuously due to the low cost of sensing and the development of storage technology. Meanwhile, the variety of data has increased as well [43, 46, 47, 48, 49, 50]. For example, the surveillance system is usually turned on 24/7 to record videos for security purpose. Digital devices such as camera and smartphones create terabytes of images and videos everyday. Billions of pieces of content are shared over social media such as Facebook and Twitter. Scalability is one of the key challenges in data clustering, i.e., traditional clustering algorithms are usually not scalable for analyzing large data sets. The growth of digital data in both the volume and variety requires data clustering techniques to efficiently transform the raw data into valuable information.

Although the major focus of this thesis is information fusion in clustering (clustering ensemble), at the beginning of this chapter we briefly review the literature of data clustering and discuss current challenges in clustering. Next, we briefly review existing clustering ensemble algorithms and the challenges of clustering ensemble. Finally, we propose two semi-supervised clustering ensemble algorithms (SEA) that utilize training data (labels) to improve clustering accuracy.

3.1 Clustering

3.1.1 Overview

Classification and clustering are two typical learning tasks that enable computer to automatically classify data observations into groups. Classification is the task of identifying the category membership for a new data observation, when a set of data observations is available for training and their category membership is known. Therefore, classification belongs to supervised learning and its goal is to formulate a mapping function between a training set and its known labels, which is used for mapping new data. Clustering is the task of grouping data observations into groups (clusters) so that data points from the same cluster are expected to be more similar to each other than to a data point from another cluster. The goal of clustering is to explore the natural structure of the given data set. In contrast to classification, clustering is often considered as unsupervised learning because the cluster labels are data driven and not pre-defined.

Suppose the goal of clustering is to divide a set of N data observations into K clusters (usually K is unknown). Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N\}$ denote the set of N data observations and $C = \{C_1, \ldots, C_k, \ldots, C_K\}$ denote the set of K clusters. Data observations/points are commonly represented as multi-dimensional vectors, each dimension of which represents a single feature (measurement, attribute, variable), i.e., data point $\mathbf{x}_i$ is represented by an F-dimensional vector $\mathbf{x}_i = [x_{i1}, \ldots, x_{if}, \ldots, x_{iF}]$. Features of data observations can be categorized into quantitative and qualitative features. Quantitative features can be measured as continuous values (such as height and weight), discrete values (such as the number of apples), and interval values (such as the duration of an event). Qualitative features include nominal features (unordered such as "shape") and ordinal features (such as "quiet" and "loud").

Clustering algorithms can be divided into hard clustering techniques and soft clustering techniques. The hard clustering algorithms assign a cluster label $\lambda_i \in \{1, \ldots, K\}$ to data point $\mathbf{x}_i$. The label assignments for all data points in $\mathbf{X}$ are represented by a label vector $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_i, \ldots, \lambda_N]^T$. In hard clustering, each data point belongs to one and only one cluster. On the other hand, the soft clustering algorithms assign a fractional degree of membership λ_{ik} to data point $\mathbf{x}_i$ for cluster C_k. The fuzzy label assignments are represented by a matrix $\boldsymbol{\Lambda} = (\lambda_{ik}) \in N \times K$. In soft clustering, each data point belongs to more than one cluster and it belongs to every cluster to a certain degree. Soft clustering is useful when the boundaries among the clusters are not well separated and ambiguous.

Different cluster validity indexes have been defined to evaluate the accuracy of clustering algorithms based on either internal or external criterion. Internal cluster validity indexes assess the fit between clustering result and the data (using only the data themselves), such as Dunn index, Davies-Bouldin index and Silhouette coefficients. External indexes measure the performance by matching a clustering to the a priori information (such as ground truth of cluster labels), such as F-measure, NMI measure, macro-precisions and micro-precisions [22, 23, 51, 52].

3.1.2 Clustering Algorithms

Clustering algorithms can be broadly categorized into two groups: hierarchical clustering and partitional clustering. The hierarchical methods produce a hierarchical structure of the given data set and cut the structure at a certain level to obtain the clusters, while the partitional methods directly partition data observations into a certain number of clusters. Many clustering algorithms have been proposed from different approaches, such as hierarchical, graph-theoretic, squared-error based and mixture-densities based clustering [22, 23, 53, 54, 55].

Hierarchical Clustering

The hierarchical structure of the given data set is usually constructed according to the proximity (similarity) matrix and described by a dendrogram (an example is shown in Figure 3.1). The top of the dendrogram represents a single cluster containing all the data points and the bottom represents singleton clusters each containing only one data point. The hierarchical methods recursively find nested clusters either in agglomerative (bottom-up) or divisive (top-down) mode. The agglomerative approach starts from singleton clusters, merges the most similar pair of clusters, and stops until

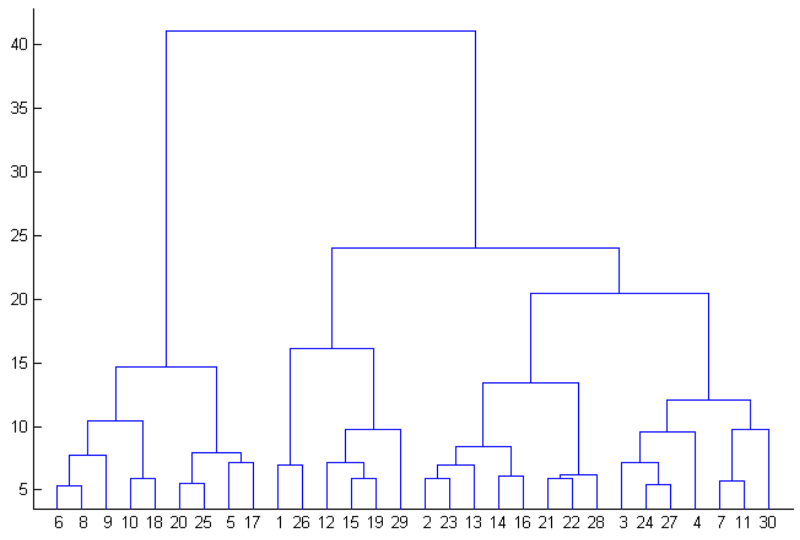


Figure 3.1: A sample dengrogram

forming one big cluster containing all data points. The divisive approach starts from the cluster containing all data points and recursively divides each cluster into smaller clusters until each cluster becomes singleton.

General steps of hierarchical agglomerative clustering algorithm are listed in Table 3.1. According to different definitions of distance between a pair of clusters, many hierarchical algorithms exist in the literature. The *single-link* and *complete-link* are the most popular algorithms [22, 23]. In the *single-link* method, distance between two clusters C_i and C_j is defined as the distance between two closest points, one from C_i and the other from C_j. In the *complete-link* method, distance between C_i and C_j is determined by two farthest points.

The advantage of hierarchical methods is that it does not require prior information about the number of clusters, since different clusterings are obtained by cutting the dendrogram at different levels. The disadvantages of hierarchical methods include:

Table 3.1: Hierarchical agglomerative clustering

1. Start with N singleton clusters. Calculate the proximity matrix for the N clusters.
2. Search the minimal distance

   $$D(C_i, C_j) = \min_{1 \le m,l \le N, m \ne l} D(C_m, C_l)$$

   where $D(*, *)$ is the distance function and combine cluster C_i and C_j to form a new cluster.
3. Update the proximity matrix by computing the distances between the new cluster and the other clusters.
4. Repeat steps 2-3 until all objects are in the same cluster.

(1) they are not capable of correcting a previous misclassification, since a data point is not re-considered once it has been assigned to a cluster; (2) it is not efficient for high-volume data sets because of the high computational cost.

Graph-Theoretic Clustering

Graph-theoretic clustering methods approach the problem by viewing data points to be clustered as the nodes of a weighted graph. The edge weight of a pair of nodes is given by the similarity measure between the corresponding data points. The hierarchal methods are related to graph-theory based clustering. The *single-link* algorithm is equivalent to seeking maximally connected sub-graphs, while the *complete-link* algorithm is equivalent to seeking maximally complete sub-graphs [22, 23].

Chameleon proposed in [56] is a hierarchical clustering algorithm based on the

k-nearest neighbor graph. The algorithm starts with dividing the graph into a large number of sub-graphs (clusters) with the minimal edge cut and then merges small subsets based on their similarity measure: relative interconnectivity and closeness. The relative interconnectivity of two clusters is calculated by normalizing the sum of weights of edges connecting the two clusters with respect to their internal interconnectivity, while the relative closeness is obtained by normalizing the average edge weights with respect to their internal closeness. The internal interconnectivity and closeness are the sum and average, respectively, of weights of edges crossing a min-cut bisection that splits the cluster into two roughly equal parts.

CLICK proposed in [57] is an another example of graph theory based clustering algorithm. It is based on calculating the minimum weight cut to form clusters. In this method, the edge weight between a pair of nodes is proportional to the probability that these two nodes belong to the same cluster. Under the assumption that the similarity values within clusters and between clusters follow Gaussian distributions with different means and variances, the edge weight between nodes (data points) i and j is determined by

$$e_{ij} = \log \frac{p_0 \sigma_B}{(1-p_0)\sigma_W} + \frac{(S_{ij}-\mu_B)^2}{2\sigma_B^2} - \frac{(S_{ij}-\mu_W)^2}{2\sigma_W^2} \tag{3.1}$$

where S_{ij} represents the similarity measure between nodes i and j, and μ_B, σ_B^2, μ_W, σ_W^2 are the means and variances of between-cluster similarities and within-clusters similarities, respectively. These parameters can be estimated from the prior knowledge or using parameter estimation methods. The algorithm recursively checks the current sub-graph containing the unclustered data points. If the sub-graph contains only one vertex, the vertex is moved to the singleton set R. If the sub-graph satisfies the stopping criterion, it is declared as a kernel. Otherwise the sub-graph is split into two parts according to a minimum weight cut. To generate the final clustering, the algorithm uses the kernels as the basic clusters and carries out a series of singleton adoptions. At the end, a merging step is performed to merge similar clusters and followed by a last singleton adoption step.

Squared-error Based Clustering

The partitional methods usually accompany the optimization of a criterion function. The most popular and simplest partitional algorithm is *K-means* [43]. It is based on minimizing the squared error between the empirical mean of a cluster and the points in the corresponding cluster, i.e.,

$$J = \sum_{k=1}^{K} \sum_{i=1}^{N_k} ||\mathbf{x}_i^{(k)} - \mathbf{m}_k||^2, \tag{3.2}$$

where $\mathbf{x}_i^{(k)}$ represents the i-th data point in the k-th cluster, N_k is the number of data points in the k-th cluster and $\mathbf{m}_k$ is the centroid (mean) of the k-th cluster. The major steps of the *K-means* algorithm is summarized in Table 5.3 [22, 23]. It starts with an initial partition of the data set with K clusters and repeatedly assigns data points to clusters in order to reduce the squared error in Eq. (3.2) until a termination criterion is met (e.g., there is no reassignment of any data point from one cluster to another, or the maximum allowed number of iterations is reached).

K-means algorithm is efficient for partitioning large data sets and identifying isolated and compact spherical clusters. However, the problem accompanied is the choice of initial partitions and the choice of the number of clusters (K). Variants of *K-means* have been developed to address the disadvantages of the original version.

Table 3.2: *K-means* clustering

1. Initialize K randomly selected data points as the cluster centers
2. Assign each data point to the closest cluster center
3. Recalculate the cluster centers using the current cluster memberships
4. Repeat steps 2-3 until the termination criterion is met

ISODATA proposed in [58] does not require the number of clusters. It estimates the number of clusters by merging and splitting clusters according to user-specified thresholds. The splitting operation eliminates the possibility of elongated clusters. The *global K-means* algorithm proposed in [59] starts from $k = 1$ and uses the centroid of the data set as the optimal position of the first cluster and adds a new cluster for each increment of k. At the k-th iteration, by considering the $k-1$ centroids determined from previous iterations (optimally) plus one of the N data points as the initial centroids, the algorithm performs N runs of *K-means* algorithm and selects the best solution as the k optimal centroids for the next iteration. Although the algorithm does not depend on the initial selection of the centroids, it requires more computational power since for each value of k it executes *K-means* N times. The robust *K-medoid* methods [60] use median of the data instead of mean. It is useful when the computation of means is not available (e.g., data consists of binary features and/or categorical features).

The *fuzzy c-means* algorithm proposed in [61] is a popular algorithm that produces soft clustering based on minimizing the square-error. The algorithm organizes the

Table 3.3: Fuzzy c-means clustering

1. Select appropriate values for c, m and ϵ and randomly initialize matrix $\mathbf{\Lambda}^{(0)}$ and set step variable $t = 0$
2. Compute $\hat{\mathbf{m}}_k{}^{(t)}$ using

$$\hat{\mathbf{m}}_k = \frac{\sum_{i=1}^{N} (\hat{\lambda}_{ki})^m \mathbf{x}_i}{\sum_{i=1}^{N} (\hat{\lambda}_{ki})^m} \tag{3.3}$$

for $k = 1, \ldots, c$

3. Compute an updated matrix $\hat{\mathbf{\Lambda}}^{(t+1)} = (\lambda_{ki}^{(t+1)})$ using

$$\hat{\lambda_{ki}} = \Big(\sum_{j=1}^{c} \Big(\frac{||\mathbf{x}_i - \mathbf{m}_k||}{||\mathbf{x}_i - \mathbf{m}_j||} \Big)^{1/m-1} \Big)^{-1} \tag{3.4}$$

for $k = 1, \ldots, c$ and $i = 1, \ldots, N$

4. Terminate if $||\hat{\mathbf{\Lambda}}^{(t+1)} - \hat{\mathbf{\Lambda}}^{(t)}|| < \epsilon$. Otherwise set $\hat{\mathbf{\Lambda}}^{(t)} = \hat{\mathbf{\Lambda}}^{(t+1)}$ and return to step 2.

data points to be clustered into c fuzzy clusters (denoted by $\mathbf{\Lambda} = (\lambda_{ki})$) by minimizing

$$J = \sum_{k=1}^{c} \sum_{i=1}^{N} (\lambda_{ki})^m ||\mathbf{x}_i - \mathbf{m}_k||^2, \tag{3.5}$$

where λ_{ki} represents the membership coefficient of data point $\mathbf{x}_i$ in the k-th cluster; $\mathbf{m}_k$ is the vector representing the center of the k-th cluster; and $m \in [1, \infty)$ is the weighting exponent. The summary of the fuzzy c-means algorithm is summarized in Table 3.3 [22, 23].

Mixture-densities Based Clustering

The mixture-densities based clustering algorithms are developed under the following assumptions: (1) data points to be clustered are drawn from one of K probability distributions (K is known), (2) the prior probability $P(C_k)$ of cluster C_k, $k = 1, \ldots, K$, is known, (3) the forms for the cluster-conditional probability densities $p(\mathbf{x}_i|C_k, \boldsymbol{\theta}_k)$ are known, where $\boldsymbol{\theta_k}$ is the unknown parameter vector of the k-th probability density function. Suppose data point $\mathbf{x}_i$ is drawn from a mixture density of

$$p(\mathbf{x}_i|\mathbf{\Theta}) = \sum_{k=1}^{K} p(\mathbf{x}_i|C_k, \boldsymbol{\theta}_k) P(C_k), \tag{3.6}$$

where $\mathbf{\Theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K]$ and $\sum_{k=1}^{K} P(C_k) = 1$. Therefore, the likelihood of the observed data set $\mathbf{X}$ is the joint density

$$p(\mathbf{X}|\mathbf{\Theta}) = \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{\Theta}). \tag{3.7}$$

The maximum-likelihood (ML) estimate $\hat{\boldsymbol{\Theta}}$ is the estimated value of $\boldsymbol{\Theta}$ that maximizes Eq. (3.7). The expectation-maximization algorithm is the most popular method to approximate the ML estimates. Theoretically, the mixtures can be constructed with any distribution functions. However, the same families with different parameters are usually considered in practice. *MCLUST* proposed in [62] considers the component density is multivariate Gaussian with a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$ as the parameters to be estimated.

The theoretical foundation for the mixture-densities based clustering methods is very well understood and these methods perform well on the data sets with moderate number of dimensions. However, these methods have the tendency to impose structure on the data which may not be there. Multivariate Gaussian distribution is often considered as a parametric model. However, high-dimensional data is usually non-Gaussian. Moreover, the number of parameters to be estimated increases rapidly when the dimensionality of the data set increases. The estimation of the unknown parameters become more complex computationally.

3.1.3 Current Challenges

In the literature of clustering, the term cluster has been used in an intuitive sense. There is no universal agreement on the formal definition of cluster [22, 23, 43, 53, 63]. A common statement about cluster is that data points in the same cluster are similar and data points from different clusters are dissimilar. However, how to measure the similarity or dissimilarity not only depends on the definition of cluster but also depends on the feature types and scales. For example, the similarity measure in K-means algorithm is based on Euclidean distance. Features measured in relatively large scales are dominant in the calculation of distance between data points.

In addition, many clustering algorithms have been proposed in the literature [22, 23, 53, 54, 55]. Clustering algorithms usually make assumptions on the data model and impose a structure on the data explicitly or implicitly. Different clustering algorithms usually produce different clusterings on the same data. If the imposed structure of a particular clustering algorithm matches the structure of given data, this clustering algorithm could produce a good clustering. There is no universal clustering algorithm that could perform well on all data sets. Due to the lack of the prior information about data structure, a variety of clustering methods are usually applied to the given data set. The selection of an appropriate clustering algorithm with some suitable parameter settings depends on the data properties and often requires domain knowledge [22, 23].

Besides, the number of clusters (K) of the given data is usually unknown in practice. Many clustering algorithms require an input of K to produce cluster labels for the given data points. Therefore, the initial guess on the number of clusters impacts the performance of clustering algorithms.

Furthermore, the volume and dimensionality of data sets to be analyzed nowadays become higher and higher. It requires the clustering algorithms to be very conscious of scaling issues. Also it is possible to pre-process data using dimensionality reduction techniques in order to decrease the computational complexity and improve the performance. However the selection of a suitable dimensionality reduction technique depends on the data and requires domain knowledge.

3.2 Clustering Ensemble

3.2.1 Overview

As mentioned earlier, selecting an appropriate clustering algorithm for a particular clustering task is a challenging problem in cluster analysis, since different clustering algorithms produce quite different clusterings for the same data set. Motivated by the improvement of classification accuracy in supervised ensemble methods, many unsupervised clustering ensemble methods have been proposed in the literature [24, 26, 27, 28, 29, 30, 31, 32, 43, 64]. By combining a set of clusterings, clustering ensemble algorithms produce a consensus clustering that is expected to be better in some sense than each individual clustering.

In general, combining multiple clusterings is more difficult than combining local decisions due to many reasons. One of the obvious reasons is that the number and shape of clusters depend on the clustering algorithms that generate them as well as their parameter settings. Another reason is that the desired number of clusters is often unknown due to the lack of prior information about the data set. Furthermore, the most important reason comes from the correspondence problem of multiple clusterings due to the fact that cluster labels are symbolic. It is possible that the same cluster labels from different clusterings represents two distinct clusters. For example, clusterer $\phi^{(1)}$ in Figure 3.2 represents a mapping function between seven data points and its cluster labels, i.e. $\boldsymbol{\lambda}^{(1)} = [1, 2, 2, 1, 3, 2, 3]^T$. Clusterer $\phi^{(2)}$ represents another mapping function which produces $\boldsymbol{\lambda}^{(2)} = [2, 1, 1, 3, 2, 3, 2]^T$. The cluster label 1 in clustering $\boldsymbol{\lambda}^{(1)}$ represents the cluster formed by $\{\mathbf{x}_1, \mathbf{x}_4\}$. The same cluster label 1 in clustering $\boldsymbol{\lambda}^{(2)}$ represents the cluster formed by $\{\mathbf{x}_2, \mathbf{x}_3\}$, different from the cluster of

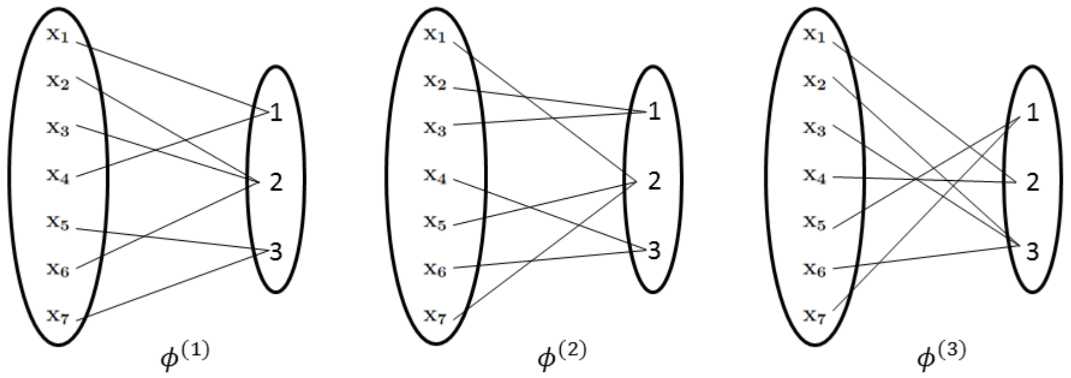


Figure 3.2: Example of clusterers: different mapping functions between data set $\mathbf{X}$ and its cluster labels $\boldsymbol{\lambda}$

$\boldsymbol{\lambda}^{(1)}$ with the same label. On the other hand, it is also possible to have two distinct label vectors that represent the same partition of a given data set. For example, clusterer $\phi^{(3)}$ shown in Fig. 3.2 represents another mapping between data points and cluster labels, i.e., $\boldsymbol{\lambda}^{(3)} = [2, 3, 3, 2, 1, 3, 1]^T$. Although $\boldsymbol{\lambda}^{(1)}$ and $\boldsymbol{\lambda}^{(3)}$ are two different label vectors, they actually represents two identical partitions of data set $\mathbf{X}$.

Recall that $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N\}$ denotes the data set of N data observations. Each data point $\mathbf{x}_i$, $i = 1, \ldots, N$, comes from the F-dimensional feature space and is represented by a row vector $\mathbf{x}_i = [x_{i1}, \ldots, x_{if}, \ldots, x_{iF}]$. Thus, data set $\mathbf{X}$ can be viewed as an $N \times F$ data matrix. Similarly, the set of D clusterings $\mathbf{\Phi} = \{\boldsymbol{\lambda}^{(1)}, \ldots, \boldsymbol{\lambda}^{(j)}, \ldots, \boldsymbol{\lambda}^{(D)}\}$, where $\boldsymbol{\lambda}^{(j)}$ represents the j-th clustering of data set $\mathbf{X}$, can be viewed as an $N \times D$ cluster label matrix. The entry on the i-th row and j-th column corresponds to the cluster label of data point $\mathbf{x}_i$ according to the j-th clustering. One approach to obtain the consensus clustering is to combine the set of base clusterings using some consensus (fusion) function. By viewing cluster labels as new data features, another approach to obtain the consensus clustering is to apply any clustering algorithm on the new feature space of cluster labels.

3.2.2 Clustering Ensemble Algorithms

Clustering ensemble methods usually consist of two major steps: base clustering generation and consensus fusion. The set of base clusterings can be generated in different ways: (1) applying different clustering algorithms, (2) applying the same clustering algorithm with different parameter settings or initializations, (3) applying the same clustering algorithm (or different clustering algorithms) to different combinations of the features and etc. Different methods of consensus fusion have been proposed in the literature and can be broadly divided into different categories, such as relabeling and voting based, co-association based, hypergraph based and mixture-densities based clustering ensemble algorithms [27, 28, 55].

Relabeling and Voting Based Clustering Ensemble

The correspondence problem is one major issue that makes the clustering ensemble difficult to solve. Thus, a direct ensemble approach is to resolve the correspondence problem and to obtain the consensus labels by a voting process. In order to vote, the number of clusters in each base clustering is required to be the same as the desired number of clusters in the final consensus clustering. The relabeling and voting based ensemble methods are easy to understand and implement. However, the correspondence problem can only be solved with certain accuracy, if all the base clusterings have the same number of clusters. Therefore, these methods are preferred when the number of clusters in all the base clusterings are the same.

The bagging method proposed in [29] is used to generate and aggregate multiple clusterings and to reduce variability in the clustering results via averaging. The proposed *BagClust1* algorithm is based on plurality voting, similar to the method used in supervised classification ensemble [65]. The summary of the *BagClust1* algorithm is listed in Table 3.4. A clustering algorithm is repeatedly applied on the bootstrap sample of the original data set with a fixed value of K. The clustering of each bootstrap sample is relabeled by permuting the cluster labels so that there is maximum overlap with the clustering of the original data set. The final consensus clustering is produced by plurality voting.

The *Voting-Merging* method proposed in [31] consists of a voting procedure and a merging procedure. Similar to bagging, the corresponding problem is also resolved by permuting the cluster labels of each base clustering according to a reference (one of the base clustrings) so that two clusters with highest percentage of common points have the same cluster label. In the voting step, the fraction of times data point $\mathbf{x}_i$ that is assigned to the j-th cluster is recorded by the i-th row and j-th column of matrix $\mathbf{\Lambda}$, denoted as λ_{ij}. The final consensus label for data point $\mathbf{x}_i$ is the one which has been assigned to this point most often, i.e., $\lambda_i = \arg\max_j \lambda_{ij}$. The output of the algorithm can either be the soft clustering $\mathbf{\Lambda}$ or the hard clustering $\boldsymbol{\lambda}$. In the merging

Table 3.4: The *BagClust1* algorithm

1. Apply clustering algorithm to the original data set $\mathbf{X}$ and obtain $\boldsymbol{\lambda}^{(0)}$
2. Form bootstrap sample $\mathcal{L}^j = \{\mathbf{x}_1^j, \ldots, \mathbf{x}_N^j\}$
3. Apply the clustering algorithm (using the same K in step 1) to obtain $\boldsymbol{\lambda}^j$
4. Permute the cluster labels of $\boldsymbol{\lambda}^j$ so that there is maximum overlap with the original clustering $\boldsymbol{\lambda}^{(0)}$ of these data points and generate base clustering $\boldsymbol{\lambda}^{(j)}$
5. Repeat step 2-4 D times and form the set of base clusterings $\boldsymbol{\Phi} = [\boldsymbol{\lambda}^{(0)}, \boldsymbol{\lambda}^{(1)}, \ldots, \boldsymbol{\lambda}^{(D)}]$
6. Assign a bagged cluster label for each data point by majority vote and obtain the consensus clustering $\boldsymbol{\lambda}$

procedure, the authors define $n(k,j) = \text{mean}_{\mathbf{x}_i \in C_k} \lambda_{ij}$ as the measure of how strong the points of cluster C_k belong to cluster C_j. If $n(k,j) = \max_{j \neq k} n(k,j)^2$, then C_j is the closest cluster to C_k. If C_j is the closest cluster to C_k and verse versa, then C_j and C_k are merged together. Additionally, a set of clusters $\{C_{k_1}, \dots, C_{k_n}\}$ is merged if C_{k_i}, $i = 1, \dots, n-1$, is the closest to its consecutive $C_{k_{i+1}}$ and the last cluster C_{k_n} is the closest to C_{k_1}.

Co-association Based Clustering Ensemble

The co-association based methods, sometimes referred to as pair-wise approach, transform the set of base clusterings into the co-association matrix, which can be viewed as a new similarity measure between data points to be clustered. The consensus clustering is obtained by applying any clustering algorithm on this new similarity measure. The objective of such a transformation is to avoid the correspondence problem. The i, j-th element of the co-association matrix $\mathbf{A}$ is determined by

$$a_{ij} = \frac{N_{ij}}{D} \tag{3.8}$$

where N_{ij} is the number of times for which data points $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same cluster among all of the D base clusterings. Since the constructed co-association matrix is $N \times N$, it is not efficient to apply the co-association based method to high-volume data sets. The number of base clusterings needs to be sufficiently large to form a meaningful co-association matrix.

The *Evidence-Accumulation* method proposed in [30] generates the single-link dendrogram based on the co-association matrix computed for the set of base clusterings. The optimal consensus clustering $\boldsymbol{\lambda}$ is obtained by cutting the dengrogram with the highest lifetime k_{opt}. The k-cluster lifetime is defined as the range of threshold values on the dendrogram to obtain k clusters. Additionally, complete-link and other hierarchical clustering algorithms can be used to clustering the co-association matrix as well.

The cluster-based similarity partitioning algorithm (*CSPA*) proposed in [24] is another co-association based method. The co-association matrix is viewed as a connected graph. The vertices are the data points to be clustered and the edge weight of a pair of vertices (data points) is the corresponding co-associations between the pair. The consensus clustering is obtained by partitioning the graph into K subgraphs using METIS, a graph partitioning algorithm proposed in [66].

Hypergraph Based Clustering Ensemble

In graph theory, a regular graph consists of vertices and edges which connects exactly two vertices. A hypergraph is a generalization of graph that consists of vertices and hyperedges. A hyperedge not only connects two vertices but also connects any set of vertices [67].

The Meta-Clustering algorithm (MCLA) proposed in [24] is a hypergraph based ensemble method. By viewing data points as vertices and clusters as hyperedges, the set of clusterings is represented as a hypergraph. The algorithm groups and collapses related clusters and assigns each data point to the collapsed cluster in which it participates most strongly. Suppose partition matrix $\mathbf{H}^{(j)} = [\mathbf{h}_1, \ldots, \mathbf{h}_{K^{(j)}}]$ represents the $K^{(j)}$ clusters of the clustering $\boldsymbol{\lambda}$. Each column of $\mathbf{H}^{(j)}$ represents one cluster and the sum of each row equals one. For hard clustering, any data point belongs to only one cluster. For a set of D clusterings, it forms a block partition matrix

Table 3.5: Meta-clustering algorithm (MCLA)

1. Construct Meta-graph by viewing all the hyperedges (columns of $\mathbf{H}$) as vertices of another regular undirected graph. Edge weights are calculated by

$$w_{a,b} = \frac{\mathbf{h}_a^\dagger \mathbf{h}_b}{||\mathbf{h}_a||_2^2 + ||\mathbf{h}_b||_2^2 - \mathbf{h}_a^\dagger \mathbf{h}_b}$$

2. Cluster Hyperedges by partitioning the meta-graph into K-balanced meta-clusters $C_q^{(M)}$, for $q = 1, \ldots, K$, each of which represents a group of matching hyperedges

3. Collapse Meta-cluster $C_q^{(M)}$ into a single hyepredge by averaging all hyperedges in $C_q^{(M)}$ to form its association vector and repeat for $q = 1, \ldots, K$

4. Compete for objects by assigning each data point to its most associated meta-cluster

$\mathbf{H} = (\mathbf{H}^{(1)} \ldots \mathbf{H}^{(D)})$ by combining all partition matrices $\mathbf{H}^{(j)}$. The total number of columns in the matrix $\mathbf{H}$ is $\sum_{j=1}^{D} K^{(j)}$.

The summary of the algorithm is shown in Table 3.5. To construct the regular undirected meta-graph, the columns of partition block matrix $\mathbf{H}$ are viewed as hyperedges, which are the vertices of the meta-graph. Edge weights of the meta-graph are proportional to the similarities between vertices. For vertices $\mathbf{h}_a$ and $\mathbf{h}_b$, the edge weight $w_{a,b}$ between them is calculated by Equation (3.9).

$$w_{a,b} = \frac{\mathbf{h}_a^\dagger \mathbf{h}_b}{||\mathbf{h}_a||_2^2 + ||\mathbf{h}_b||_2^2 - \mathbf{h}_a^\dagger \mathbf{h}_b}. \tag{3.9}$$

The calculation is based on the binary Jaccard measure, which is the ratio of the intersection to the union of the sets of data points corresponding to the two hyperedges. Therefore, two clusters with more common points are more similar so that the edge weight between the corresponding hyperedges has a larger value.

Hyperedges (clusters) are divided into groups by partitioning the meta-graph into K balanced meta-clusters. Therefore, hyperedges from the same meta-cluster are more similar to each other than to hyperedges from a different meta-cluster. Since each vertex in the meta-graph represents a distinct cluster, a meta-cluster consists of a group of most related clusters. Meta-clusters are collapsed into a single meta-hyperedge. Each meta-hyperedge has an association vector which contains an entry for each object describing its level of association with the corresponding meta-cluster. The association vector is calculated by averaging all hyperedges of a particular meta-cluster. The final consensus clustering is obtained by assigning to each data point its most associated meta-cluster.

Mixture-densities Based Clustering Ensemble

The set of base clusterings can be viewed as a new space of the given data set transformed from the original feature space. The mixture-densities based approach is based on proposing a probability model of the consensus clustering in the new space. The consensus clustering is the solution to the maximum likelihood problem for a given set of base clusterings. The likelihood function is optimized with respect to the parameters of a finite mixture distribution.

In [32], the authors assumed that the labels in $\mathbf{y}_i$ (the i-th row of label matrix

$\mathbf{\Phi}$ contains all the labels assigned to data point $\mathbf{x}_i$) are modeled as random variables drawn from a probability distribution described as a mixture of multivariate component densities:

$$P(\mathbf{y}_i|\mathbf{\Theta}) = \sum_{k=1}^{K} \alpha_k P_k(\mathbf{y}_i|\boldsymbol{\theta}_k), \tag{3.10}$$

where each component corresponds to one of the K consensus clusters (with prior probabilities $P(C_k) = \alpha_k$ for $k = 1, \ldots, K$) and is parametrized by $\boldsymbol{\theta}_k$. By assuming all the $\mathbf{y}_i$ (for $i = 1, \ldots, N$) are independent and identically distributed, the logarithmic likelihood function to the parameters $\mathbf{\Theta} = \{\alpha_1, \ldots, \alpha_K, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K\}$ given the set $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ is given by

$$\log L(\mathbf{\Theta}|\mathbf{Y}) = \log \prod_{i=1}^{N} P(\mathbf{y}_i|\mathbf{\Theta}) = \sum_{i=1}^{N} \log \sum_{k=1}^{K} \alpha_k P_k(\mathbf{y}_i|\boldsymbol{\theta}_k). \tag{3.11}$$

The combination of a set of base clusterings into a consensus clustering is now formulated as a maximum likelihood estimation problem, i.e., the best fitting mixture density for a given $\mathbf{Y}$ is described by

$$\hat{\mathbf{\Theta}} = \arg\max_{\mathbf{\Theta}} \log L(\mathbf{\Theta}|\mathbf{Y}). \tag{3.12}$$

The log-likelihood function in Eq. (3.11) can be optimized using the EM algorithm by assuming the existence of a set of unobserved (hidden) latent data $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$. The EM algorithm starts with arbitrary parameters $\mathbf{\Theta}' = \{\alpha'_1, \ldots, \alpha'_K, \boldsymbol{\theta}'_1, \ldots, \boldsymbol{\theta}'_K\}$. The E-step calculates the expected values of the hidden variables and the M-step maximized the likelihood by calculating new and better parameter estimates. The consensus clustering is obtained by assigning to data point $\mathbf{x}_i$ the consensus label which corresponds to the largest expected values among the hidden variables in $\mathbf{z}_i$.

In [26], the authors proposed a mixture model that generates the set $\mathbf{Y}$ in the Bayesian setting. That means a membership to multiple consensus clusters is allowed in this model. Let $\boldsymbol{\theta}_i$ denote the latent mixed-membership vector for $\mathbf{y}_i$. The authors assumed that $\boldsymbol{\theta}_i$ is sampled from a Dirichlet distribution with parameter α and each component of hidden vector $\mathbf{z}_i$ is sampled from a discrete distribution of $\boldsymbol{\theta}_i$. The full generative process for each data point $\mathbf{x}_i$ is summarized in Table 3.6. Given the model parameter α and $\beta = \{\beta_{hj}\}$ where $h = 1, \ldots, K$ and $j = 1, \ldots, M$, the joint distribution of latent and observed variables $\{\mathbf{x}_i, \mathbf{z}_i, \boldsymbol{\theta}_i\}$ is calculated by

$$P(\mathbf{x}_i, \mathbf{z}_i, \boldsymbol{\theta}_i|\alpha, \beta) = P(\boldsymbol{\theta}_i|\alpha) \prod_{j=1, \exists x_{ij}}^{M} P(z_{ij} = h|\boldsymbol{\theta}_i) p(x_{ij}|\beta_{hj}), \tag{3.13}$$

where $\exists x_{ij}$ denotes that there exists a j-th base clustering result for $\mathbf{x}_i$. The marginal probability for each $\mathbf{x}_i$ can be computed by integrating over the latent variables $\{\mathbf{z}_i, \boldsymbol{\theta}_i\}$, i.e.,

$$p(\mathbf{x}_i|\alpha, \beta) = \int_{\boldsymbol{\theta}_i} p(\boldsymbol{\theta}_i|\alpha) \prod_{j=1, \exists x_{ij}}^{M} \sum_{h} p(z_{ij} = h|\boldsymbol{\theta}_i) p(x_{ij}|\beta_{hj}) d\boldsymbol{\theta}_i. \tag{3.14}$$

The parameters are estimated by variational inference method.

3.3 Semi-supervised Clustering

As discussed earlier, supervised learning requires training data (labeled) such as classification, while unsupervised learning does not requires any training data such as

Table 3.6: Generative process for Bayesian model

1. Choose $\boldsymbol{\theta}_i \sim \text{Dirichlet}(\alpha)$
2. For the j-th base clustering:
    (a) Choose a component $z_{ij} = h \sim \text{discrete}(\boldsymbol{\theta}_i)$
    (b) Choose the base clustering result $x_{ij} = h \sim \text{discrete}(\beta_{hj})$

clustering. Besides these two learning techniques, there exists a learning considered as halfway between supervised and unsupervised learning, called semi-supervised learning [68]. It can be viewed as unsupervised learning guided by the constrains (i.e. labellings provided by experts). Usually in unsupervised learning no data observations are labeled. For semi-supervised learning, in addition to unlabeled data, labels are provided for a portion of the data as supervision information. On the other hand, semi-supervised learning can also be viewed as supervised learning with additional information. For a supervised method, the learning process is performed on the training data only, while in the semi-supervised learning, additional information is provided by the unlabeled data as well. If the additional information is useful in the inference of training data, then the semi-supervised learning can yield an improvement over supervised learning.

Although no training data (labeled) is required for the traditional cluster analysis, in many real applications a small portion of labeled data is usually available. Side information such as a pair-wise constraint may be very useful in finding the hidden structure of the given data. Therefore, there is a growing interest in semi-supervised clustering that utilizes labeled data to improve the clustering results [69, 70, 71, 72, 73, 74]. The semi-supervised clustering algorithm proposed in [69] is based on modifying existing clustering objective function. The *Constrained-K-means* proposed in [70] is a semi-supervised $K - means$ that uses the seed clustering (user-specified labeled data) to initialize the algorithm. The *Boosting* algorithm proposed in [73] is an iterative algorithm using must-link and cannot-link constraints to improve the performance of any clustering algorithm.

3.4 Semi-supervised Clustering Ensemble

We have discussed earlier in this chapter that clustering results can be improved by either clustering ensemble or semi-supervised clustering techniques. The success of these approaches motivates us to combine the benefit of both techniques to further improve clustering results. The side supervision information can be provided in two steps. One option is to provide the supervision information in the base clustering generation step, i.e., applying semi-supervised clustering algorithms to generate the set of base clusterings. The voting-based semi-supervised ensemble method proposed in [75] is formulated in this way. Another option is to utilize the supervision information in the fusion step. The algorithm proposed later in this section approaches the problem in this way. It calculates the association between each data point and the training clusters and relabels the local cluster assignments according to the training clusters. In the context of this thesis, the generation of base clusterings is based on unsupervised clustering algorithms and the fusion of base clusterings is guided by the side information. Thus, we name the proposed algorithms as the semi-supervised clustering ensemble algorithms (SEA). They both consist of two major steps: base clusterings generation and fusion. The base clustering generation step is common to existing ensemble methods. For the base clustering fusion step, we propose two consensus fusion algorithms: SHSEA and HHSEA.

3.4.1 Soft-to-hard Semi-supervised Clustering Ensemble Algorithm (SHSEA)

Suppose that input data set $\mathbf{X}$ is the combination of a training set $\mathbf{X}_r$ and a testing set $\mathbf{X}_u$. The training set $\mathbf{X}_r$ contains data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N_r}\}$, for which labels are provided in a label vector $\boldsymbol{\lambda}_r$. The testing data set $\mathbf{X}_u$ contains data points $\{\mathbf{x}_{N_r+1}, \ldots, \mathbf{x}_N\}$, the labels of which are unknown. The consensus cluster label vector (output of SEA) of testing set $\mathbf{X}_u$ is denoted by $\boldsymbol{\lambda}_u$. The size of training set $\mathbf{X}_r$ is measured by the number of data points in the training set and denoted by N_r, i.e., $|\mathbf{X}_r| = N_r$. Similarly, the size of testing set $\mathbf{X}_u$ is measured by the number of data points in the testing set and denoted by N_u, i.e., $|\mathbf{X}_u| = N_u$. According to the training and testing sets, the label matrix $\boldsymbol{\Phi}$ can be partitioned into two block matrices $\boldsymbol{\Phi}_r$ and $\boldsymbol{\Phi}_u$, each of which contains all the labels corresponding to the data points in the training set $\mathbf{X}_r$ and testing set $\mathbf{X}_u$ respectively. Suppose training data points belong to K_0 classes and all training points from the k-th class form one cluster, denoted by C_r^k $(k = 1, \ldots, K_0)$. Therefore, the training set $\mathbf{X_r}$ consists of a set of K_0 clusters $\{C_r^1, \ldots, C_r^k, \ldots, C_r^{K_0}\}$. If the size of cluster C_r^k is denoted by N_r^k, the total number of training points equals to the sum of N_r^k, i.e., $N_r^k = \sum_{k=1}^{K_0} N_r^k$. We rearrange label matrix $\boldsymbol{\Phi}_r$ to form K_0 block matrices: $\boldsymbol{\Phi}_r^1, \ldots, \boldsymbol{\Phi}_r^k, \ldots, \boldsymbol{\Phi}_r^{K_0}$. Each block matrix $\boldsymbol{\Phi}_r^k$ contains the base cluster labels of data points in the k-th training cluster C_r^k.

The fusion idea of SHSEA is stated as follow: (1) for a particular data point count the number of agreements between its label and the labels of training points in each training cluster, according to an individual base clustering, (2) calculate the association vector between this data point and the corresponding base clustering, (3) compute the average association vector by averaging the association vectors between this data point and all base clusterings, and (4) repeat for all data points and derive the soft consensus clustering for the testing set. The summary of SHSEA is provided in Table 3.7. Since the overall consensus cluster labels are derived from the fuzzy (soft) label matrix, we name this approach as the soft-to-hard semi-supervised clustering ensemble algorithm (SHSEA).

According to the j-th clustering $\boldsymbol{\lambda}^{(j)}$, we compute the association vector $\mathbf{a}_i^{(j)}$ for the i-th unlabeled data point $\mathbf{x}_i$, where $i = 1, \ldots, N_u$ and $j = 1, \ldots, D$. Since there are K_0 training clusters, the association vector $\mathbf{a}_i^{(j)}$ has K_0 entries. Each entry describes the association between data point $\mathbf{x}_i$ and the corresponding training cluster. The k-th entry of the association vector $\mathbf{a}_i^{(j)}$ is measured as the occurrence of cluster label of data point $\mathbf{x}_i$ among the labels of reference data points in the k-th training cluster (according to base clustering $\boldsymbol{\lambda}^{(j)}$), i.e.,

$$\mathbf{a}_i^{(j)}(k) = \frac{\text{occurrence of } \boldsymbol{\Phi}_u(i,j) \text{ in } \boldsymbol{\Phi}_r^k(:,j)}{N_r^k}, \tag{3.15}$$

where $\boldsymbol{\Phi}_u(i,j)$ represents the cluster label of data point $\mathbf{x}_i$ and $\boldsymbol{\Phi}_r^k(:,j)$ represents the labels of reference points in the k-th training category generated according to base clustering $\boldsymbol{\lambda}^{(j)}$. In order to fuse the set of base clusterings, the weighted average association vector $\mathbf{a}_i$ of data point $\mathbf{x}_i$ is computed by averaging D association vectors

Table 3.7: Soft-to-hard semi-supervised clustering ensemble algorithm (SHSEA)

---

* Input: Base clusterings $\mathbf{\Phi}$, Reference label vector $\boldsymbol{\lambda}_r$, Weight Vector $\boldsymbol{\omega}$

* Output: Consensus label matrix $\mathbf{\Lambda}_u$ or label vector $\boldsymbol{\lambda}_u$

(a) According to label vector $\boldsymbol{\lambda}_r$, rearrange base clusterings $\mathbf{\Phi}$ into $K_0 + 1$ sub-matrices $\{\mathbf{\Phi}_r^1, \ldots, \mathbf{\Phi}_r^k, \ldots, \mathbf{\Phi}_r^{K_0}, \mathbf{\Phi}_u\}$

(b) For data point $\mathbf{x}_i$, calculate the k-th element of the association vector $\mathbf{a}_i^{(j)}$ by

$$\mathbf{a}_i^{(j)}(k) = \frac{\text{occurrence of } \mathbf{\Phi}_u(i,j) \text{ in } \mathbf{\Phi}_r^k(:,j)}{N_r^k}$$

and repeat for $k = 1, \ldots, K_0$ to form the association vector $\mathbf{a}_i^{(j)}$

(c) Compute the weighted average association vector $\mathbf{a}_i$ for data point $\mathbf{x}_i$ by $\mathbf{a}_i = \sum_{j=1}^{D} \omega_j \mathbf{a}_i^{(j)}$, where ω_j is the weight of j-th base clustering

(d) Compute the association level γ_i of data point $\mathbf{x}_i$ to all training clusters by $\gamma_i = \sum_{k=1}^{K_0} \mathbf{a}_i(k)$.

(e) Compute the membership information of data point $\mathbf{x}_i$ to every cluster by normalizing $\mathbf{a}_i$

(f) Repeat step (b) to (d) to generate the association level vector $\boldsymbol{\gamma}_u$ and repeat step (b) to (e) to generate the soft consensus label matrix $\mathbf{\Lambda}_u$

(g) Based on the average association vector $\mathbf{a}_i$, assign data point $\mathbf{x}_i$ its most associated cluster id

(h) Repeat (g) for all $i = 1, \ldots, N_u$ to form the hard consensus label vector $\boldsymbol{\lambda}_u$

$\mathbf{a}_i^{(j)}$, i.e.,

$$\mathbf{a}_i = \sum_{j=1}^{D} \omega_j \mathbf{a}_i^{(j)}, \tag{3.16}$$

where ω_j is the corresponding weight of the j-th local clusterer. Note that the weight vector $\boldsymbol{\omega} = [\omega_1, \ldots, \omega_j, \ldots, \omega_D]$ is either known or can be estimated. When local clusterers are equally important, the weights are all the same, i.e., $\omega_j = 1/D$. Each entry of $\mathbf{a}_i$ describes the overall association between data point $\mathbf{x}_i$ and the corresponding training cluster. As a consequence, the summation of all the entries of $\mathbf{a}_i$ could be used to describe the association between data point $\mathbf{x}_i$ and all the training clusters quantitatively. We define it as the association level of data point $\mathbf{x}_i$ to all training clusters and denote it as γ_i, i.e.,

$$\gamma_i = \sum_{k=1}^{K_0} \mathbf{a}_i(k), \tag{3.17}$$

where k is the k-th entry of $\mathbf{a}_i$ and corresponds to the k-th reference cluster. By computing the association level for all data observations, the association level vector $\boldsymbol{\gamma}_u$ for the testing set $\mathbf{X}_u$ is made up by stacking association level γ_i for all $i = 1, \ldots, N_u$, i.e., $\boldsymbol{\gamma}_u = [\gamma_1, \gamma_2, \ldots, \gamma_{N_u}]^T$. We have two options to present the overall consensus clustering for testing set $\mathbf{X}_u$. One option is to produce a soft consensus label matrix $\boldsymbol{\Lambda}_u$. The i-th row of $\boldsymbol{\Lambda}_u$ is computed by normalizing the average association vector $\mathbf{a}_i$, i.e.,

$$\boldsymbol{\Lambda}_u(i,:) = \mathbf{a}_i^T / \gamma_i. \tag{3.18}$$

The other option is to produce a hard consensus label vector $\boldsymbol{\lambda_u}$. The consensus cluster label assigned to each data point is its most associated category labels in the corresponding average association vector. Since the overall hard cluster labels are assigned according to the soft label matrix, we name this algorithm as the soft-to-hard semi-supervised clustering ensemble algorithm (SHSEA). The normalized soft consensus label matrix ($\boldsymbol{\Lambda}_u$) can be used as the output of the algorithm.

3.4.2 Hard-to-hard Semi-supervised Clustering Ensemble Algorithm (HHSEA)

Follow the naming convention, the other semi-supervised ensemble method is called hard-to-hard semi-supervised clustering algorithm (HHSEA), since the overall cluster labels are assigned based on hard label matrix. The fusion idea stated as follow: (1) for a particular data point count the number of agreements between its label and the labels of training points in each training cluster, according to an individual base clustering, (2) calculate the association vector between this data point and the corresponding base clustering, (3) assign this data point to its most associated cluster label (4) repeat for all data points and all base clusterings to relabel the labels in matrix $\boldsymbol{\Phi}_u$ and (5) apply majority voting to derive hard consensus clustering. The summary of this algorithm is provided in Table 3.8.

Table 3.8: Hard-to-hard semi-supervised clustering ensemble algorithm (HHSEA)

---

* Input: Base clusterings Φ, Reference label vector $\boldsymbol{\lambda}_r$

* Output: Hard clustering $\boldsymbol{\lambda}_u$

(a) According to label vector $\boldsymbol{\lambda}_r$, rearrange base clusterings $\boldsymbol{\Phi}$ into $K_0 + 1$ sub-matrices $\{\boldsymbol{\Phi}_r^1, \dots, \boldsymbol{\Phi}_r^k, \dots, \boldsymbol{\Phi}_r^{K_0}, \boldsymbol{\Phi}_u\}$

(b) For data point $\mathbf{x}_i$, calculate the k-th element of the association vector $\mathbf{a}_i^{(j)}$ by

$$\mathbf{a}_i^{(j)}(k) = \frac{\text{occurrence of } \boldsymbol{\Phi}_u(i,j) \text{ in } \boldsymbol{\Phi}_r^k(:,j)}{N_r^k}$$

and repeat for $k = 1, \dots, K_0$ to form the association vector $\mathbf{a}_i^{(j)}$

(c) Assign data point $\mathbf{x}_i$ its most associated cluster ids, which corresponds to the highest entry of association vector $\mathbf{a}_i^{(j)}$

(d) According to the j-th clustering, repeat step (b) and (c) for all data points

(e) Repeat (b) - (d) for $j = 1, \dots, D$ and relabel $\boldsymbol{\Phi}_u$ into $\boldsymbol{\Phi}'_u$

(f) Apply majority voting on $\boldsymbol{\Phi}'_u$ to derive hard consensus clustering $\boldsymbol{\lambda}_u$

Chapter 4

Distributed Clustering System

Motivated by the improvement in the detection performance of the distributed detection system, we propose a similar distributed clustering system. The input to the system is data set **X** and it is usually pre-processed before entering to the system, especially when it is high-dimensional and high-volume. Data pre-processing is a necessary step to improve the results and reduce the computational complexity of cluster analysis [76, 77]. The basic structure of the proposed distributed clustering system consists of a base clustering generator (produces multiple base clusterings) and a fusion center (combines base clusterings into a consensus clustering). The output of the system is the consensus clustering, which is expected to be more accurate than each individual base clustering.

In this chapter we first briefly introduce the proposed distributed clustering system. Next, we discuss several common data pre-processing techniques used in clustering. In Section 4.3 we design four base clustering generators based on different ways of generating the base clusterings. In Section 4.4 we design two operational modes of the fusion center according to the absence and presence of training data. Based on the soft-to-hard semi-supervised clustering ensemble method (SHSEA), we propose a new cluster detection algorithm in Section 4.5. It is implemented in the proposed system to provide feedback when data observations from a new class is detected. Finally in Section 4.6, we propose a parallel distributed clustering system, consisting of multiple distributed clustering systems, to efficiently partition high volume data.

4.1 Overview of the System

Similar to the distributed detection system, the proposed distributed clustering system consists of multiple local clusterers and a fusion center, as shown in Figure 4.1. The term clusterer is used to denote the local processing unit that produces cluster labels for the given data set $\mathbf{X}$. The number of local clusterers is usually given a priori and is determined by the users based on the specification of the application. A set of base clusterings $\{\boldsymbol{\lambda}^{(1)}, \ldots, \boldsymbol{\lambda}^{(j)}, \ldots, \boldsymbol{\lambda}^{(D)}\}$ is received in the fusion center, where they are combined into a consensus clustering $\boldsymbol{\lambda}_u$. We will discuss the local clusterers and the fusion center in details later in this chapter.

As discussed earlier, there are different possibilities for generating multiple base clusterings: applying different clustering algorithms in local clusterers, applying the same clustering algorithm with different initializations and clustering using different features of the given data [24, 29, 30, 43]. In the fusion center, any unsupervised clustering ensemble algorithms (reviewed in Section 3.1.2) can be implemented to analyze data sets without training observations. In this scenario the fusion center is denoted as the unsupervised fusion center. On the other hand, any semi-supervised clustering ensemble methods (proposed in Section 3.2.2) can be implemented to analyze data sets containing labeled training observations. In this scenario fusion center is denoted

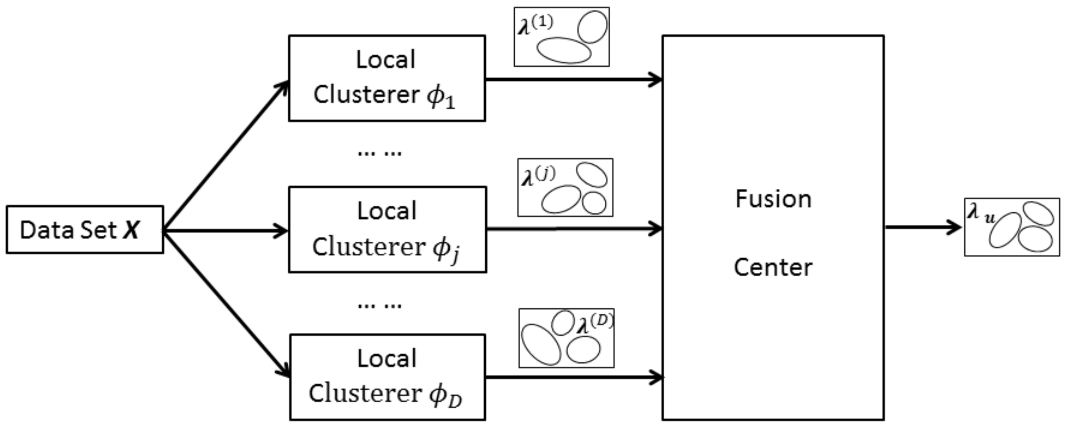


Figure 4.1: Distributed clustering system

as the semi-supervised fusion center.

4.2 Data pre-processing

In this section, we briefly discuss three data pre-processing techniques: normalization, feature extraction and feature selection. In our proposed distributed clustering system, these techniques can be applied to the input data before it is sent to each local clusterer. The major objective of data pre-processing is to improve the accuracy of clustering results and to reduce the computational complexity.

4.2.1 Normalization

In practice many data sets contain features that are measured in different units and scales. Features measured in relatively large scales may play a dominant role in the similarity measure and influence the accuracy of the clustering results [78, 79]. As a consequence, normalizing such features is an important pre-processing procedure, especially when the similarity measure is based on Euclidean distances. For example, consider a data set containing age, height and weight information of 1000 elementary students. The age feature is measured in years and varies from 3 to 15. The height feature is measured in centimeters and varies from 80 cm to 180 cm. The weight features is measured in kilograms and varies from 25 kg to 65 kg.

The common normalization methods used in clustering include min-max normalization, Z-score normalization [78, 79]. Min-max normalization is a linear transformation of features into a specified range, which equalizes the magnitude of features and prevents overweighting features measured in relatively large scale over features measured in relatively small scale. Z-score normalization maps each original feature into a zero mean and unit variance feature [80, 81]. The selection of normalization methods is data dependent and the range features to be normalized to is often determined by domain experts.

Min-max Normalization

Suppose $\mathbf{x}^{(f)}$ represents the f-th feature of data set $\mathbf{X}$. Let $\mathbf{x}^{(f)}_{max}$ and $\mathbf{x}^{(f)}_{min}$ represent the maximum and minimum value of the f-th feature respectively. Min-max normalization maps the f-th feature into range [0, 1] by

$$\mathbf{x}^{(f)}_{Norm} = \frac{\mathbf{x}^{(f)} - \mathbf{x}^{(f)}_{min}}{\mathbf{x}^{(f)}_{max} - \mathbf{x}^{(f)}_{min}}. \tag{4.1}$$

If a specific range [a, b] is preferred instead of [0, 1], Eq. (4.1) can be replaced by

$$\mathbf{x}^{(f)}_{Norm} = a + \frac{(\mathbf{x}^{(f)} - \mathbf{x}^{(f)}_{min})(b-a)}{\mathbf{x}^{(f)}_{max} - \mathbf{x}^{(f)}_{min}}. \tag{4.2}$$

Z-score Normalization

Suppose the mean and standard deviation of feature $\mathbf{x}^{(f)}$ is denoted by $\mu^{(f)}$ and $\sigma^{(f)}$. Z-score normalization maps the f-th feature into a zero mean and unit variance feature by

$$\mathbf{x}^{(f)}_{Norm} = \frac{\mathbf{x}^{(f)} - \mu_f}{\sigma_f}. \tag{4.3}$$

4.2.2 Feature Extraction

In many applications, data sets to be analyzed consist of hundreds and even thousands of features. Analysis of high-dimensional data usually requires high computational capability and storage capacity. Moreover, due to the fact that features are often statistical measurements of the underlying phenomenon, it is highly possible that features of certain dimensions are correlated and not as important as others for understanding the whole data. Therefore, dimensionality reduction is a necessary step in cluster analysis.

Feature extraction is a common technique to reduce the data dimensionality. The objective of feature extraction is to transform the set of original features/variables to a set of uncorrelated principal variables. The transformation of data variables could be linear, such as principal component analysis and the Karhunen-Loève transformation [76, 82]. Many nonlinear dimensionality reduction techniques also exist, such as the techniques proposed in [83, 84, 85]. In this thesis, we focus on the linear transformation - principal component analysis (PCA).

PCA is a mathematical tool to transform a set of correlated variables into a new set of uncorrelated and orthogonal variables (principal components). In order to keep the accuracy of cluster analysis, the variation present in the original data set needs to be retained as much as possible. Therefore, the principal components are the linear combinations of the original variables which retain the variance of the original data in the way that the first principal component contains the largest variance; second principal component is orthogonal to the first principal component and contains the second largest variance; and the rest principal components behave in the same way. The dimensionality of the data set could be reduced by neglecting the principal components with relatively small variances [82].

Let $\vec{X} = [X_1, X_2, \ldots, X_F]^T$ denote a random vector of F random variables with mean vector $\mathbb{E}(\vec{X}) = [\mu_1, \mu_2, \ldots, \mu_F]^T$ and covariance matrix $\text{Var}(\vec{X}) = \mathbf{\Sigma}$. Suppose $Z = v_1X_1 + v_2X_2 + \cdots + v_FX_F$ is an arbitrary linear combination of $X_1, X_2, \ldots, X_F$ with coefficients $v_1, v_2, \ldots, v_F$, i.e., $Z = \mathbf{v}^T\vec{X}$, where $\mathbf{v}^T = [v_1, v_2, \ldots, v_F]$ and $\text{Var}(Z) = \text{Var}(\mathbf{v}^T\mathbf{X}) = \mathbf{v}^T\Sigma\mathbf{v}$. The problem to find the principal components Z_f $(f = 1, \ldots, F)$ that retain the variance of original variables is equivalent to find $\mathbf{v}_f$, the coefficients of linear combination $Z_f = \mathbf{v}_f^T\vec{X}$, such that $\text{Var}(Z_f)$ is maximal subject to the normalization $\mathbf{v}_f^T\mathbf{v}_f = 1$. The coefficient vector $\mathbf{v}_1$ for the first principal component is the largest characteristic root of the covariance matrix $\mathbf{\Sigma}$. The coefficient vector $\mathbf{v}_F$ for the F-th component is the smallest characteristic root.

Suppose the contribution to the total variation of the first L principal components $Z_1, \ldots, Z_L$ is calculated by

$$\eta_L = \frac{\sum_{q=1}^{L} \text{Var}(Z_q)}{\sum_{q=1}^{F} \text{Var}(Z_q)}. \tag{4.4}$$

If the ratio η_L in Eq. (4.4) is large enough, the study about the variability in $\vec{X}$ may be confined to study about these first L principal components. Thus, the dimensionality of the data set could be reduced from F to L, where L is the number of principal components that contain most of the variation of the data set $\mathbf{X}$. Although the transformed variables retain most of the variation of the original variables, they alter the representation of the data and sometimes make it difficult to understand the clustering representation in the new space [76].

4.2.3 Feature Selection

Feature selection is another way to reduce the dimensionality of the given data set. It refers to the process of selecting a subset of the original features and it is broadly categorized into filter and wrapper approaches. Usually the filter methods are based on determining the feature dependency and relevance [86, 87, 88], while the wrapper methods are based on optimizing a well-specified objective function [89, 90, 91]. In this thesis, we focus on filter methods because they are usually easier and faster to implement. We briefly review the algorithm proposed in [86] which selects features based on the feature similarity measure. It is an unsupervised feature selection method which does not require any training data points and the corresponding cluster labels. For the scenarios when training data is available, we propose a semi-supervised feature selection algorithm based on feature ranking.

Unsupervised Feature Selection

The feature selection algorithm proposed in [86] has two major steps. The first step is to partition the original feature set into several subsets based on the k-nearest neighbor principal using maximum information compression index λ_2 as the feature similarity measure, i.e., for two random variables x and y,

$$\begin{aligned} 2\lambda_2(x,y) &= \mathrm{Var}(x)+\mathrm{Var}(y) \\ &- \sqrt{(\mathrm{Var}(x)+\mathrm{Var}(y))^2-4\mathrm{Var}(x)\mathrm{Var}(y)(1-\rho(x,y)^2)}. \end{aligned} \tag{4.5}$$

When x and y are linearly dependent, $\lambda_2 = 0$. The value of λ_2 increases as the amount of dependency between x and y decreases. The measure is sensitive to scaling of the

Table 4.1: Unsupervised feature selection using feature similarity

```
* Input: Data set $\mathbf{X}$
* Output: Data set $\mathbf{X}_{fs}$
    (a) Compute the $k$-nearest features for each feature
    (b) Select the feature having the most compact subset (compactness is determined by its distance to the farthest neighbor)
    (c) Discard the $k$-neighboring features in the subset
    (d) Repeat until all of the features are either selected or discarded
```

variables and invariant to translation and rotation of the variables. These properties makes λ_2 suitable as the feature similarity measure.

The second step is to select a feature from each subset to represent the data. The summary of the unsupervised feature selection algorithm is listed in Table 4.1.

Semi-supervised Feature Selection

Feature ranking is a common feature selection method since it is simple, scalable and independent of the choice of clustering algorithm [92, 93]. For a given training data set $\mathbf{X}_r$ and its true class assignments $\boldsymbol{\lambda}_r$, a scoring function $R(f)$ is computed according to the f-th feature and class assignments $\boldsymbol{\lambda}_r$. Features are sorted in the descending order of $R(f)$ and selected according to the ranking.

In this section, we propose a semi-supervised feature selection algorithm which is based on feature ranking. The scoring function $R(f)$ is modeled as the accuracy of clustering algorithm using the f-th feature. By applying the selected clustering algorithm to the combination of training and testing data points, cluster labels are generated not only for testing data points but also for the training (also called reference) data points. When the true cluster assignments for the reference data points are available, we propose to estimate the accuracy of clustering algorithm in term of micro-precision [52] by comparing the generated cluster labels and the true cluster assignments of all reference points. Suppose reference data set $\mathbf{X_r}$ contains N_r data points that belong to K_0 classes and N_r^k represents the number of reference data points in the k-th reference cluster that are correctly assigned to the corresponding class. Corresponding class here represents the true class that has the largest overlap with the k-th cluster. The micro-precision (MP) is calculated by

$$\mathrm{MP} = \sum_{k=1}^{K_0} N_r^k / N_r. \tag{4.6}$$

The summary of the proposed algorithm is listed in Table 4.2. By computing the mean and standard deviation of scoring function $R(f)$, features with higher micro-precision are selected. If a specific number F_d is provided as the input (by users), then the scoring function $R(f)$ is sorted in the descending order and the first F_d features are selected.

4.3 Base Clustering Generator

In this section, local clusterers are viewed as a black box, which takes data set $\mathbf{X}$ as the input and produces a set of base clusterings as the output. We name this black box as the base clustering generator (BCG) and denote it by $\mathbf{\Phi}$. The internal structure of base clustering generator is shown in Figure 4.2. It consists of D local

Table 4.2: Semi-supervised feature selection using feature ranking

```
* Input: Training set $\mathbf{X}_r$, training labels $\boldsymbol{\lambda}_r$, testing set
  $\mathbf{X}_u$ and $F_d$ (optional)
* Output: Data set $\mathbf{X}_{FS}$
    (a) Apply clustering algorithm to the $f$-th feature
        and compute the score
    (b) Repeat step (a) for all features
    (c) Select features whose score are greater than the
        summation of mean and standard deviation of
        $R(f)$
    (d) (optional) Sort $R(f)$ in the descending order and
        select the first $F_d$ features
```

clusterers and the number of local clusterers D is usually determined by the users based on the application specification.

In this thesis, our major interest is information fusion in cluster analysis (clustering ensemble). We apply the most popular and simple K-means algorithm in each local clusterer except otherwise stated. In practice, different clustering algorithms can be implemented in the local clusterers to generate base clusterings. The selection of clustering algorithms depends on the data to be analyzed and usually requires domain knowledge. To describe the setting of base clustering generator, we first define some necessary parameters as follow:

- $\phi^{(j)}$: j-th local clusterer
- D: total number of local clusterers in $\mathbf{\Phi}$

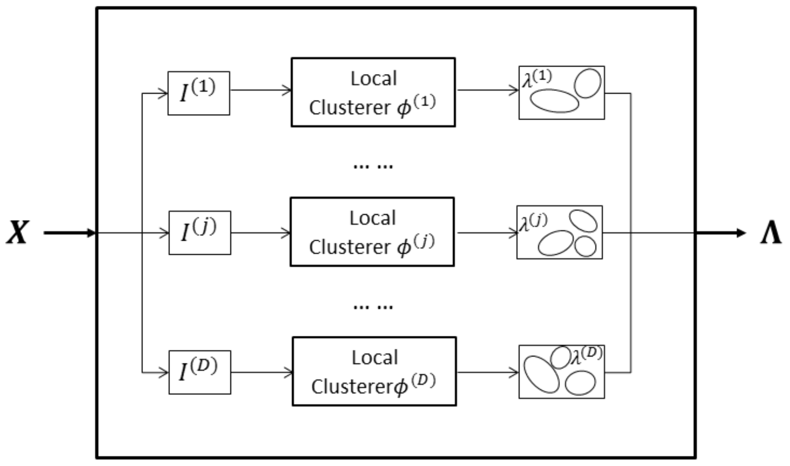


Figure 4.2: Base clustering generator with D local clusterers

- $I^{(j)}$: input of local clusterer $\phi^{(j)}$
- $\lambda^{(j)}$: output of local clusterer $\phi^{(j)}$
- $K^{(j)}$: number of clusters in $\lambda^{(j)}$

One possible way to design the base clustering generator is to build D identical local clusterers and apply the same clustering algorithm with different initializations in each local clusterer. We set $D = 21$ and denote this base clustering generator as $\mathbf{\Phi}_1$. The set of base clusterings generated by $\mathbf{\Phi}_1$ is named as "BASE1". The parameter settings of $\mathbf{\Phi}_1$ is listed in Table 4.3. In this design, the clustering processes are distributed over different local clusterers. The advantage is that each local clusterer

Table 4.3: Base clustering generators: $\mathbf{X}$ represents the input data matrix, F represents the number of features (columns) of $\mathbf{X}$, and $\mathbf{x}^{(j)}$ represents the j-th feature (column) of $\mathbf{X}$

Base Clustering Generator	Set Name	No. of Local Clusterers (D)	Local Clusterer $\phi^{(j)}$ Input ($I^{(j)}$)	Local Clusterer $\phi^{(j)}$ No. of Clusters ($K^{(j)}$)
$\mathbf{\Phi}_1$	BASE1	21	$\mathbf{X}$	$K^{(j)} = K_0$
$\mathbf{\Phi}_2$	BASE2	F	$\mathbf{x}^{(j)}$	$K^{(j)} = K_0$
$\mathbf{\Phi}_3$	BASE3	21	$\mathbf{X}$	$K^{(j)} \in [K_0, 40]$
$\mathbf{\Phi}_4$	BASE4	F	$\mathbf{x}^{(j)}$	$K^{(j)} \in [K_0, 40]$

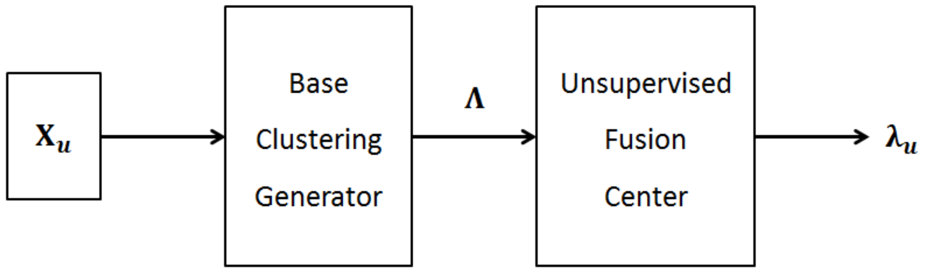
has the access to the entire data matrix and generates base clusterings based on all the information. In the literature, many clustering ensemble methods are evaluated by generating base clusterings in this way [24, 26, 29, 30, 79].

Another way to design the base clustering generator is to apply clustering algorithm to only one of data features in each local clusterer. For a data set containing F features, there are $D = F$ local clusterers in the generator. We denote this base clustering generator as $\mathbf{\Phi}_2$ and the set of base clusterings generated by this generator as "BASE2". The parameter settings of $\mathbf{\Phi}_2$ is listed in Table 4.3. In this design, data features are distributed over different local clusterers. Each local clusterer only has the access to one of the features and partitions data points from a specific aspect of the data. It is suitable for data sets whose features are measured in diverse scales. It is also suitable for data sets whose features are heterogeneous or categorical when the dissimilarity measure based on all features does not have a real meaning. It is also suitable for the scenarios when features or attributes of the data set are not shareable between organizations due to privacy, ownership or other reasons.

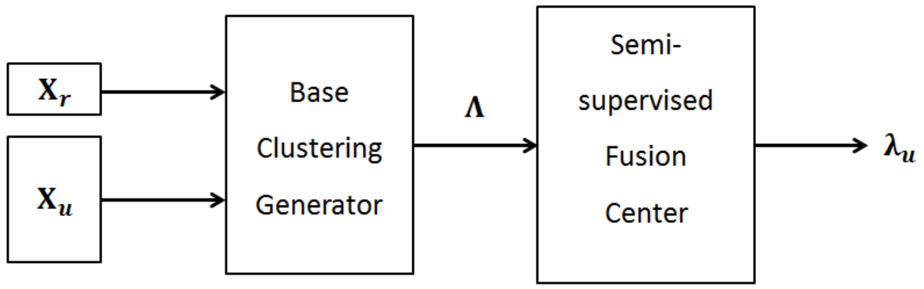
As discussed in Section 3.4 the semi-supervised clustering ensemble is expected to perform better when $K^{(j)}$ (the number of clusters in the j-th base clustering) is larger than the expected number of consensus clusters K_0. Therefore, we modify base clustering generators $\mathbf{\Phi}_1$ and $\mathbf{\Phi}_2$ by setting $K^{(j)}$ to relatively large integers. Due to the fact that the optimal value of $K^{(j)}$ is data dependent, we propose to randomly select an integer value for $K^{(j)}$ to avoid the selection of a suitable value for $K^{(j)}$. The parameter settings of the modified base clustering generators $\mathbf{\Phi}_3$ and $\mathbf{\Phi}_4$ are also listed in Table 4.3. The sets of base clusterings generated by the modified generators are denoted as "BASE3" and "BASE4" respectively.

4.4 Fusion Center

As shown in Figure 4.1, the fusion center receives the set of base clusterings (cluster labels) and produces a consensus clusterings without the access to the input data set. According to the absence and presence of the labelled data observations, we design two operational modes for the fusion center: unsupervised and semi-supervised mode. The unsupervised mode shown in Figure 4.3 (a) is designed for the scenario when no labelled training data is available. In this mode the input to the base clustering generator is the testing data set $\mathbf{X}_u$ only. We can implement the traditional unsupervised clustering ensemble algorithms in the fusion center. On the other hand the semi-supervised mode shown in Figure 4.3 (b) is designed for the scenario when labelled training data is available. In this mode the input of the system is the combination of testing set $\mathbf{X}_u$ and an additional training set $\mathbf{X}_r$ (with known cluster labels). The base clustering generator produces cluster labels for both testing and reference data points and sends all labels to the fusion center. We implement the proposed semi-supervised clustering ensemble algorithms in the fusion center.



(a) Unsupervised mode



(b) Semi-supervised mode

Figure 4.3: Two operational modes of the fusion center

4.5 System Feedback

Although training data provides some cluster information about the given data set, we may always question on whether the number of training clusters is enough or whether a new cluster is necessary to describe the nature of the given data set. It motivates us to propose a new cluster detection algorithm to automatically determine whether the additional data observations belong to existing classes or a new class. We implement it to build a new cluster detector and insert this unit into the semi-supervised mode to provide feedback to the system, as shown in Figure 4.4. When the output of the detector is "Yes", the system sends out alarm signal to request additional training data points with labels or switches to the unsupervised mode.

4.5.1 New Cluster Detection

The proposed new cluster detection algorithm is based on computing and comparing the association levels of additional data points to all training clusters and the association levels of existing data points to all training clusters. By viewing the training data points as the scatter points that clearly outline the shape of each cluster of the given data set, the testing data points are expected to form a set of clusters, similar to the clusters formed by training data points in size, shape and quantity. Intuitively, a data point belonging to the existing classes of the training data should locate inside the regions outlined by all the training points and this point should highly associate with one of the training clusters. In contrast, a data point from a new class other than the existing training classes should locate outside from the regions outlined by all the training points. The association level of this data point to all the training clusterers should be relatively low, compared with that of a data point from existing classes.

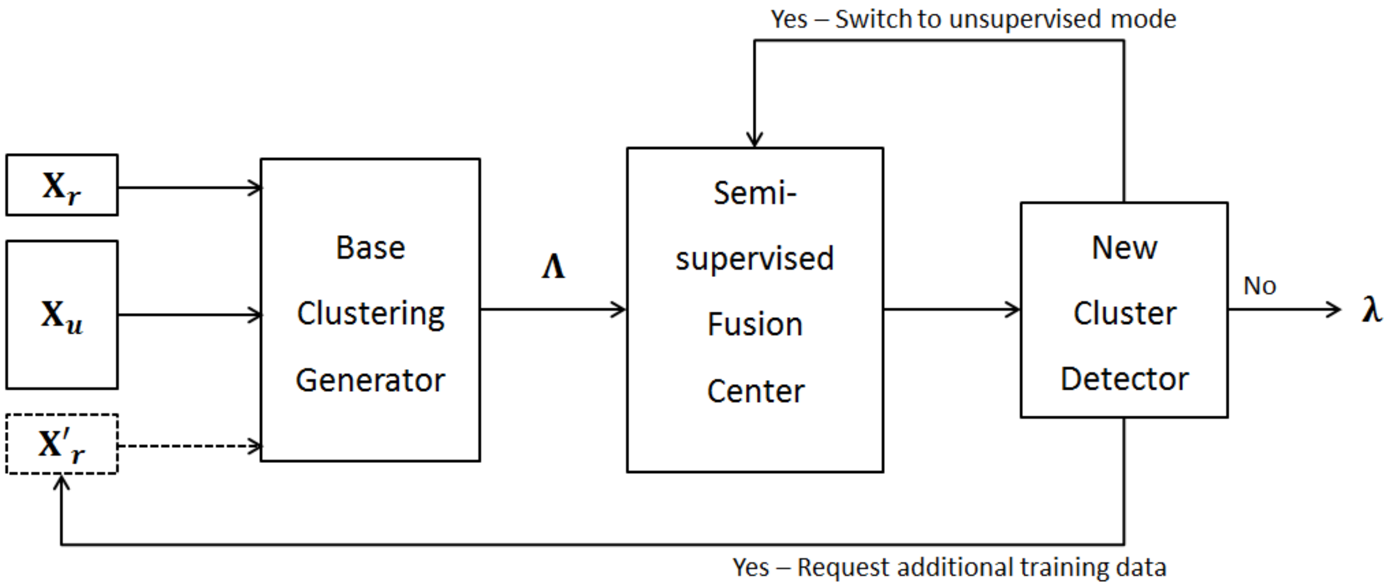

Yes – Switch to unsupervised mode
$\mathbf{X}_r$
$\mathbf{X}_u$
$\mathbf{X}'_r$
Base Clustering Generator
$\mathbf{\Lambda}$
Semi-supervised Fusion Center
New Cluster Detector
No
$\boldsymbol{\lambda}$
Yes – Request additional training data

Figure 4.4: Semi-supervised mode with feedback

Recall that in the soft-to-hard semi-supervised clustering algorithm (SHSEA), the association between data point $\mathbf{x}_i$ and each training cluster is measured quantitatively by Equation (3.15) and represented by the association vector $\mathbf{a}_i^{(j)}$. In order to obtain the consolidated clustering, the average association vector $\mathbf{a}_i$ is calculated by averaging the association vectors $\mathbf{a}_i^{(j)}$ for $j = 1, \ldots, D$, where D is the total number of base clusterings. The association level of data point $\mathbf{x}_i$ to all the training clusters is denoted by γ_i and calculated as the summation of all entries of the average association vector $\mathbf{a}_i$. As a consequence, when additional observations become available we could make decisions about the category information of the additional data points by comparing the distribution of the association level of original data points to the distribution of the association level of the additional data points. If the distribution of the association level of the additional data points is consistent with that of the original data points, the additional data points are expected to come from the existing classes. Otherwise, they are expected to come from a new class.

In Table 4.4, we list the summary of the new cluster detection algorithm. Suppose the original data set $\mathbf{X}_b$ consists of two parts: the training set $\mathbf{X}_r$ and the testing set $\mathbf{X}_u$. The additional data observations form a set of data points, denoted by $\mathbf{X}_a$. The input of the proposed algorithm is the combination of the original data set $\mathbf{X}_b$ and the additional data set $\mathbf{X}_a$, i.e., $\mathbf{X} = \{\mathbf{X}_r, \mathbf{X}_u, \mathbf{X}_a\}$. The input percentage η is selected by the user and is used to determine the threshold γ_{th} for the association level, i.e., $\gamma_{th} = \max(\gamma_b) \times \eta$. Denote the association level of the original data points and additional data points to all the training clusters by γ_b and γ_a respectively. The sizes of testing set $\mathbf{X}_u$ and additional set $\mathbf{X}_a$ are denoted by N_u and N_a respectively. The numbers of original and additional data points whose association levels are less than the threshold γ_{th} are denoted by N_b and N_{new} respectively. To determine whether a new cluster is necessary, we perform a hypothesis testing with two hypotheses:

H_0 : No data observations come from a new class

H_1 : Some data observations come from a new class.

The threshold N_{th} for the hypothesis testing is calculated by

$$N_{th} = N_a \times \frac{N_b}{N_u}. \tag{4.7}$$

When $N_{new} < N_{th}$, the hypothesis H_0 is favoured and the new cluster indicator is set to be 0. When $N_{new} \geq N_{th}$, the hypothesis H_1 is favoured and the new cluster indicator is set to be 1.

4.6 Parallel Distributed Clustering System

One of the challenges in clustering is that most of the existing clustering algorithms are not scalable. Large number of data observations not only increase the computational complexity but also decrease the accuracy of the clustering algorithms [24, 43]. Ideally an efficient way of performing cluster analysis on high-volume data sets is to divide the testing set into several smaller subsets and perform cluster analysis on each individual subset in parallel. The overall clustering can be obtained by integrating cluster labels of all subsets. This approach is often referred to as distributed clustering in the literature [79, 94, 95]. Obviously, the aforementioned correspondence problem (accompanied in clustering ensemble) is always accompanied in this approach, since

Table 4.4: New Cluster Detection Algorithm

```
* Input: Data set $\mathbf{X}$; Percentage $\eta$
* Output: New cluster indicator $i_{new}$
    (a) Apply SSEA on $\mathbf{X}_b = \{\mathbf{X}_r, \mathbf{X}_u\}$ and obtain the association level vector $\gamma_b$
    (b) Set the threshold $\gamma_{th} = \max(\gamma_b) \times \eta$
    (c) Count the number of original data points $N_b$ satisfying $\gamma_b < \gamma_{th}$
    (d) Apply SSEA on $\mathbf{X} = \{\mathbf{X}_r, \mathbf{X}_u, \mathbf{X}_a\}$ and obtain the association level vector $\gamma_a$
    (e) Count the number of additional data points $N_{new}$ satisfying $\gamma_a < \gamma_{th}$
    (f) Set the threshold for the hypothesis testing as $N_{th} = N_a \times \frac{N_b}{N_u}$ and determine $i_{new}$ by
        $i_{new} = \begin{cases} 0 & \text{if } N_{new} < N_{th} \\ 1 & \text{if } N_{new} \geq N_{th} \end{cases}$
```

the same symbolic labels from different subsets may represent two distinct clusters. Due to the fact that our proposed semi-supervised algorithms assign labels to each data point according to the training clusters, the correspondence problem could be avoid as long as all subsets share the same training set. It motivates us to proposed a modified system for clustering high-volume data sets. We named this modified system as the parallel distributed clustering system. It consists of several distributed clustering system connected in parallel and the basic structure of this parallel system is shown in Figure 4.5.

Suppose a given data set $\mathbf{X}$ contains a training set $\mathbf{X}_r$ and a testing set $\mathbf{X}_u$ and the testing set X_u is divided into Q subsets, i.e., $\mathbf{X}_u = \{\mathbf{X}_u^1, \ldots, \mathbf{X}_u^Q\}$. The input of the q-th distributed clustering system (sub-system) $\mathbf{\Psi}^q$ is the combination of training set $\mathbf{X}_r$ and the q-th subset $\mathbf{X}_u^q$, i.e., $\mathbf{X}^q = \{\mathbf{X}_r, \mathbf{X}_u^q\}$ where $q = 1, \ldots, Q$. The output of the q-th sub-system $\mathbf{\Psi}^q$ contains cluster labels of data observations in subset $\mathbf{X}_u^q$, denoted as $\boldsymbol{\lambda}_u^q$. The overall consensus clustering $\boldsymbol{\lambda}_u$ of the entire testing set $\mathbf{X}_u$ is obtained by combining Q segments of clustering label vector, i.e., $\boldsymbol{\lambda}_u = \{\boldsymbol{\lambda}_u^1, \ldots, \boldsymbol{\lambda}_u^q, \ldots, \boldsymbol{\lambda}_u^Q\}$.

The proposed parallel distributed clustering system is suitable for data sets with a large number of data observations. In such a parallel system data observations are distributed over each sub-system. All the sub-systems work individually and simultaneously. The parallel system makes clustering ensemble scalable and more efficient. In addition, in many real applications, data observations to be clustered are owned by different organizations and they are not shareable due to privacy and ownership reasons. When the same set of reference data observations are used in each sub-system, the proposed extended semi-supervised method is applicable in this scenario.

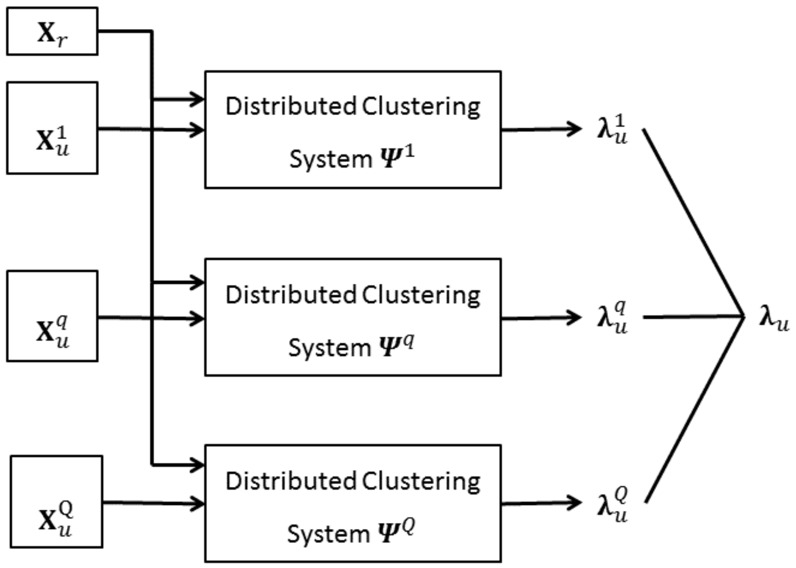

$\mathbf{X}_r$
$\mathbf{X}_u^1$
$\mathbf{X}_u^q$
$\mathbf{X}_u^Q$
Distributed Clustering System $\boldsymbol{\Psi}^1$
Distributed Clustering System $\boldsymbol{\Psi}^q$
Distributed Clustering System $\boldsymbol{\Psi}^Q$
$\boldsymbol{\lambda}_u^1$
$\boldsymbol{\lambda}_u^q$
$\boldsymbol{\lambda}_u^Q$
$\boldsymbol{\lambda}_u$

Figure 4.5: Parallel distributed clustering system

Chapter 5

Numerical Examples

In this chapter we provide numerical examples to demonstrate the performance of the proposed distributed systems. In Section 5.1, we evaluate the performance of the distributed detection system using neonatal EEG signals and provide examples to show the improved performance of the system when correlation between local decisions is considered. In Section 5.2, we perform series of experiments to evaluate the performance of distributed clustering system. We demonstrate the effect of base clusterings, normalization and dimentionality reduction on the clustering and clustering ensemble methods. We also evaluate the performance of the proposed new cluster detection algorithm and the performance of the proposed parallel distributed clustering system when the size of the input data is large.

5.1 Distributed Detection

We use data set obtained in the Neonatal Unit at McMaster's University Hospital to evaluate the performance of the distributed detection system when the local decisions are considered as correlated. The data set consists of a single channel EEG measurements sampled at 1 ms which were obtained from twenty-two neonates diagnosed with brain development issues. Consequently we expect that the number of seizures will be sufficiently large and thus sufficient for maximum likelihood (ML) estimation. A piece of EEG background signal and a piece of EEG seizure signal are shown in Figure 5.1. Since the aforementioned local detectors have different window (epoch size) properties the local decisions are properly shifted in order to be aligned in time. In addition to using spectral error criterion the data is segmented into stationary segments so that the seizure frequencies (prior probabilities) do not change significantly.

First we calculate empirical correlations in order to validate our assumption that the local decisions are actually correlated. Consequently the mean of the Pearson correlation coefficients in the absence of seizures is 0.45 with standard deviation of 0.22. Similarly in the presence of seizures the mean of the correlation coefficient is 0.68 with a standard deviation of 0.19. As expected the correlation coefficients are significantly higher in the presence of seizures. In order to evaluate the performance of the system for all the patients, we present the average results in Table 5.1 for local detectors and proposed ML-based information fusion detection with and without correlated decisions model. Obviously by fusing local detectors' decisions we achieve significant improvement especially in terms of the probability of false alarms. Note that Liu's detector still offers the best performance with respect to missed seizures.

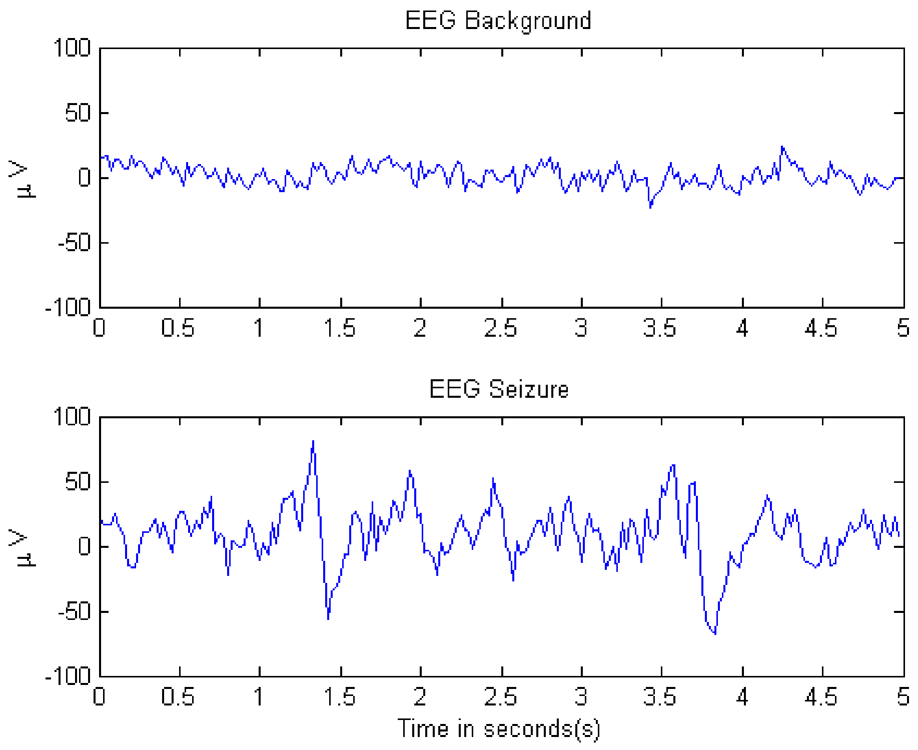


Figure 5.1: EEG background and EEG seizure signals

Table 5.1: Average neonatal seizure detection performance

	Liu	Gotman	Celka	Uncorrelated	Correlated
False alarm	0.32	0.17	0.21	0.18	0.11
Missed detection	0.04	0.29	0.27	0.06	0.04

We believe that this is mainly due to a shorter time-frame so that the weights in the fusion center are not updated with sufficient dynamic.

5.2 Distributed Clustering

In this thesis, we evaluate the performance of the proposed distributed clustering system using two types of data. One type comes from UCI machine learning repository [96], which provides hundreds of data sets for the study of classification and clustering. In the literature, many clustering and clustering ensemble algorithms were evaluated using data sets from this repository [26, 59, 79, 97]. The other type of data comes from Dr. David Andrew's laboratory (DWALab) at Sunny Brook Hospital (Toronto, Ontario). This data is used to study human breast cancer cells undergoing treatment of different drugs. The cancer cells are plated into clear-bottom well plates and 10 types of treatments are taken placed to different cells. Images of the untreated and treated cells are captured by using the high content imaging system and processed by the CAFE (Classification and Feature Extraction of micro-graphs of cells) software to extract useful information. In total 705 attributes/features per cell are recorded for further analysis [98]. The high dimensionality of this data set provides us the opportunity to evaluate the effect of dimensionality reduction techniques on clustering and clustering ensemble methods. Since this data set contains data observations from 11 different classes, we can easily create testing sets by selecting data from different combination of classes.

Since the ground truth of class assignments for each data set are available, we use micro-precision [52] as our metric to measure the accuracy of clustering result with respect to the expected (true) labelling. Recall that data set $\mathbf{X}$ contains N data points that belong to K_0 classes and N_k represents the number of data points in the k-th cluster that are correctly assigned to the corresponding class. By corresponding class we refer to the true class that has the largest overlap with the k-th cluster. The micro-precision (MP) is calculated by

$$\text{MP} = \sum_{k=1}^{K_0} N_k/N. \tag{5.1}$$

We use four UCI data sets to evaluate our proposed system. Since the evaluation of semi-supervised clustering ensemble methods requires reference data points with known labels, we divide each UCI data set into a testing set and a training set. The testing set contains data points to be clustered, while the training set contains reference data points with known labels. Similarly we create seven testing sets using DWALab data and each testing set has a corresponding training set. In Table 5.2 we list the information about these data sets, such as the number of data points, features and classes.

5.2.1 Original Data Sets

We start the evaluation process using the original data sets. Original data in this thesis refers to data that has not been processed by any normalization or dimensionality reduction techniques. Recall that we designed four base clustering generators

Table 5.2: Data Information I: the number of data points, features and classes

Data Sets	Data Points	Features	Classes
Ionosphere	270	34	2
Pima	591	8	2
Balance	482	4	3
Wine	137	13	3
DWALabSet1	300	705	2
DWALabSet2	300	705	2
DWALabSet3	300	705	2
DWALabSet4	450	705	3
DWALabSet5	450	705	3
DWALabSet6	450	705	3
DWALabSet7	600	705	4

($\mathbf{\Phi}_1$ to $\mathbf{\Phi}_4$) in Section 4.3. To study the effect of base clusterings on clustering ensemble problem, we generate four different sets of base clusterings (BASE1 to BASE4) for each data set. Note that base clustering generator $\mathbf{\Phi}_1$ is designed based on the common way used in the literature to generate base clusterings [24, 26, 29, 30]. For example, in [26] clustering ensemble methods were evaluated using 100 sets of base clusterings, each of which consists of 20 base clusterings generated by K-means with different initializations.

To evaluate different clustering ensemble methods, we apply HGPA, CSPA, MCLA (proposed in [24]) and BCE (proposed in [26]) in the unsupervised fusion center and apply SHSEA and HHSEA (proposed in Section 3.4) in the semi-supervised fusion center. Recall that the ratio of number of reference data points (N_r) to number of testing data points (N_u) is denoted by P. We set $P = 25\%$ in the experiments and repeat each experiment 100 times to calculate the average micro-precision.

The micro-precision of K-means clustering algorithm using all original features is listed in Table 5.3. The maximum and minimum micro-precision of K-means using features individually are also listed in Table 5.3. Among all 11 data sets the maximum MP of K-means using a single feature is higher than MP of K-means using

Table 5.3: Micro-precision of K-means using all features and single feature of original data

Data Sets	$K - means$		
	All Features	Single Feature	
		Max	Min
Ionosphere	0.7111	0.7667	0.5148
Pima	0.6532	0.7107	0.5094
Balance	0.5498	0.5253	0.5127
Wine	0.7007	0.7883	0.3766
DWALabSet1	0.5033	0.7917	0.5000
DWALabSet2	0.5033	0.7233	0.5000
DWALabSet3	0.5367	0.7933	0.5000
DWALabSet4	0.3400	0.5642	0.3333
DWALabSet5	0.3600	0.5578	0.3392
DWALabSet6	0.3474	0.5419	0.3357
DWALabSet7	0.2630	0.4433	0.2583

all features except data set “Balance”. Recall that BASE1 set of base clusterings is generated by repetitively applying K-means to all features together, while BASE2 is generated by applying K-means to each feature individually. Therefore, we expect the micro-precision of ensemble methods using BASE2 to be higher than that of BASE1 (except “Balance” data set), since BASE2 contains a certain number of “better” base clusterings. In addition, the performance of SHSEA using BASE2 is expected to be better than HHSEA, since base clusterings with higher MP are given larger weights in the consensus fusion step. Furthermore, recall that BASE3 (BASE4) is generated in the same way as BASE1 (BASE2) respectively except that $K^{(j)}$ (the number of clusters in each local clusterers) are set to be greater than K_0 (the expected number of clusters). Therefore, we expect the performance of SHSEA and HHSEA using BASE3 (BASE4) to be better than BASE1 (BASE2), since the proposed semi-supervised methods are expected to perform better when data points are divided into smaller groups (as explained earlier in Section 3.4).

The micro-precision of our proposed system (four unsupervised and two semi-supervised ensemble methods) using four sets of base clusterings (BASE1 to BASE4) is illustrated by sub-figure (a) in Figures 5.2 to 5.12. The performance of SHSEA and HHSEA is represented by series SH(P25) and HH(P25) respectively and $P25$ means the ratio of reference and testing points is $P = 25\%$. Among four groups of clustering results, the bar corresponding to the highest average MP of the unsupervised ensemble methods and the bars corresponding to the highest MP of SHSEA and HHSEA are labelled in each chart. It is clear that the performance of the proposed semi-supervised methods conforms with our expectations.

The winner of unsupervised methods, its corresponding winning set of base clusterings, and the winning set of SHSEA and HHSEA are summarized in Table 5.4. The winning ensemble method of each data set is highlighted in bold. Compared to the micro-precision of K-means algorithm (Table 5.3), the clustering results have been improved by both operational modes of the proposed system. The performance of the semi-supervised mode is better than the unsupervised mode (except "DWAL-abSet1"). The winning set of base clusterings is either BASE2 or BASE4 except the "Balance" data set. Note that although the actual winning set for "Ionosphere" is BASE3, the MP of SHSEA using BASE4 is very close to the winning MP. The winning ensemble method using original data sets is SHSEA.

To study the effect of quantity of reference points on semi-supervised clustering ensemble methods, we repeat the experiments in semi-supervised mode by selecting different numbers of reference points, i.e., by changing the value of P in $N_r = P \cdot N_u$. The micro-precision of SHSEA and HHSEA with different values of P is shown in sub-figures (c) and (e) of Figures 5.2 to 5.12. The highest MP of both semi-supervised

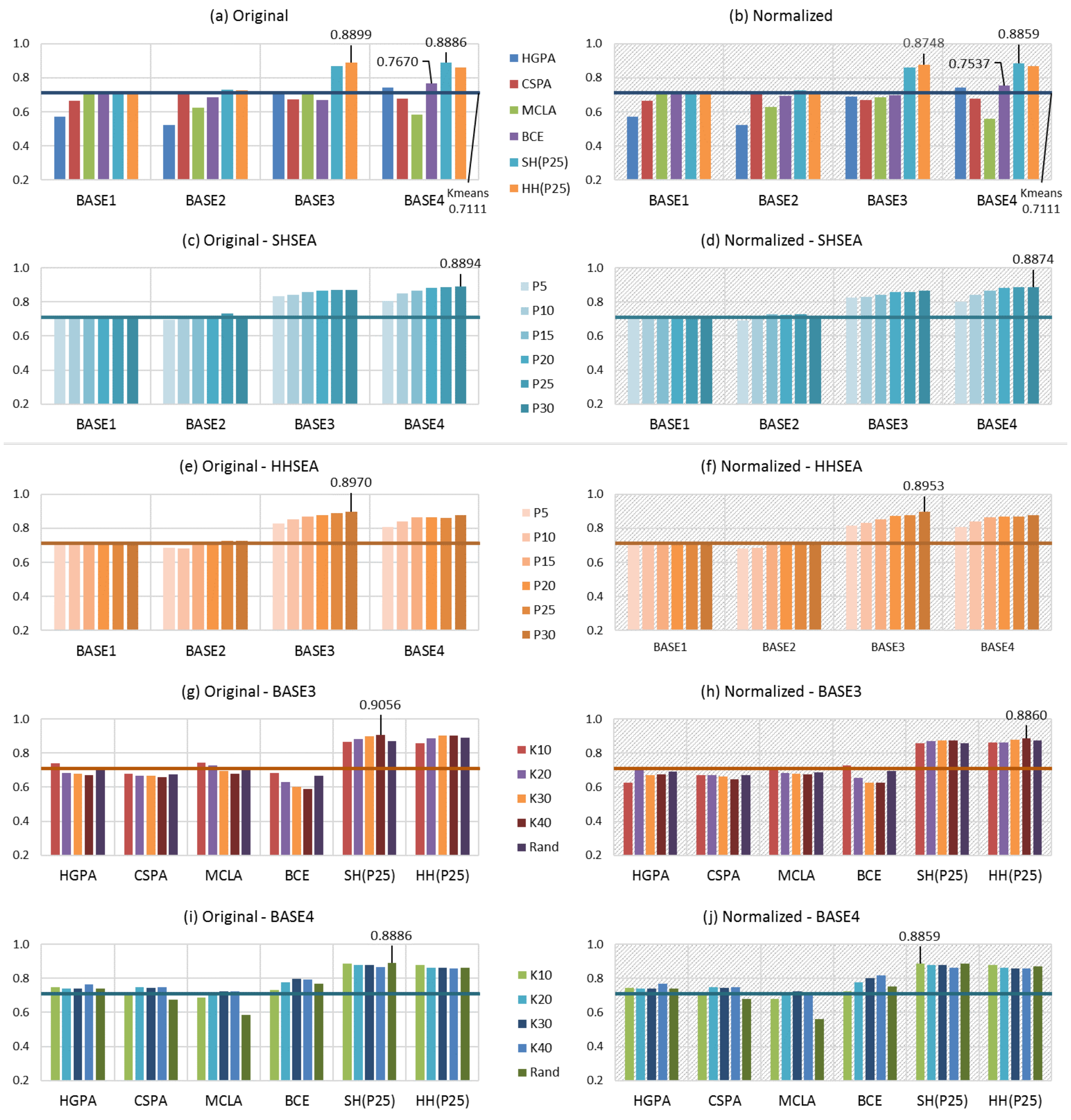


Figure 5.2: Original vs. Normalized Ionosphere

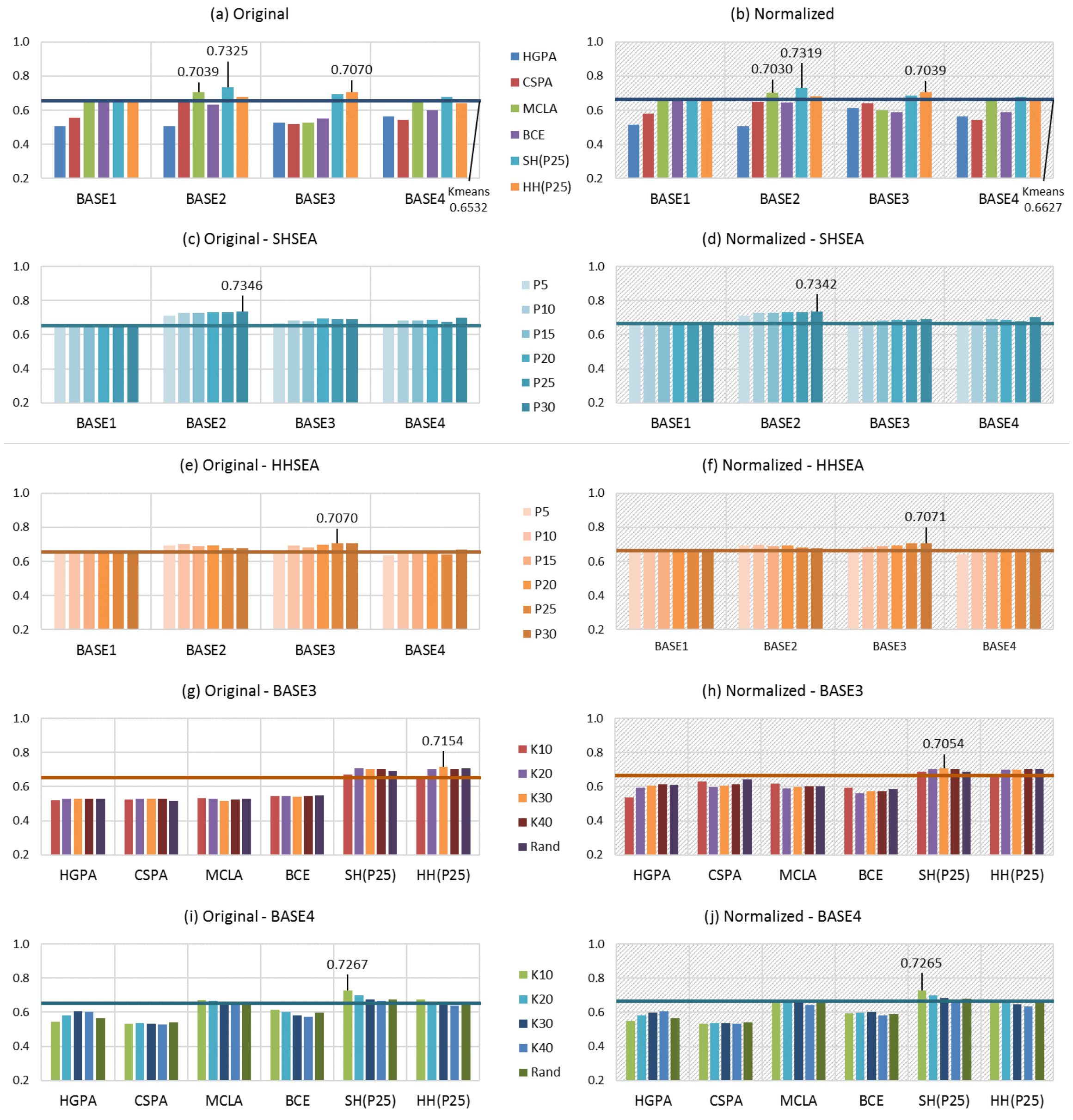


Figure 5.3: Original vs. Normalized Pima

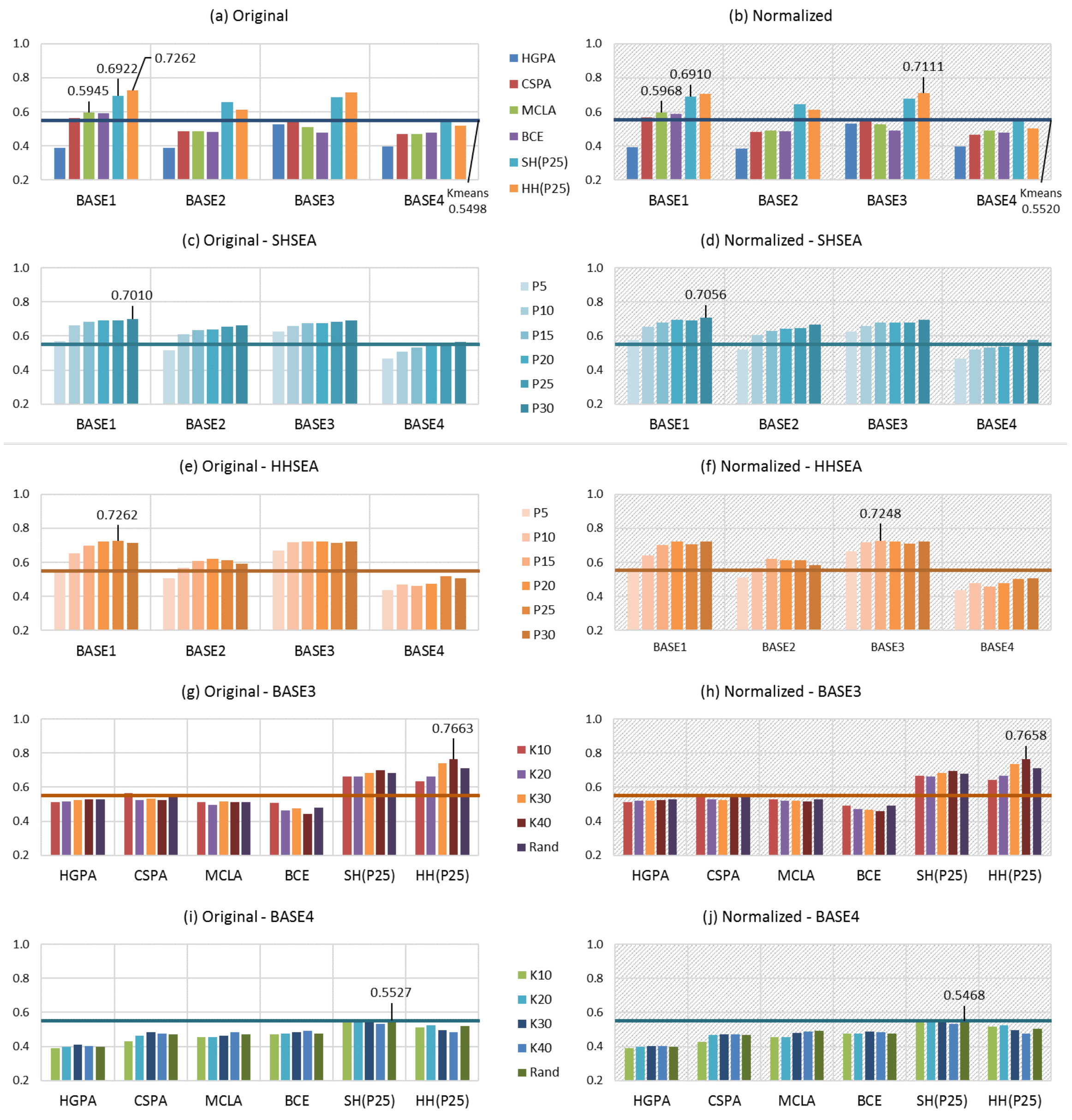


Figure 5.4: Original vs. Normalized Balance

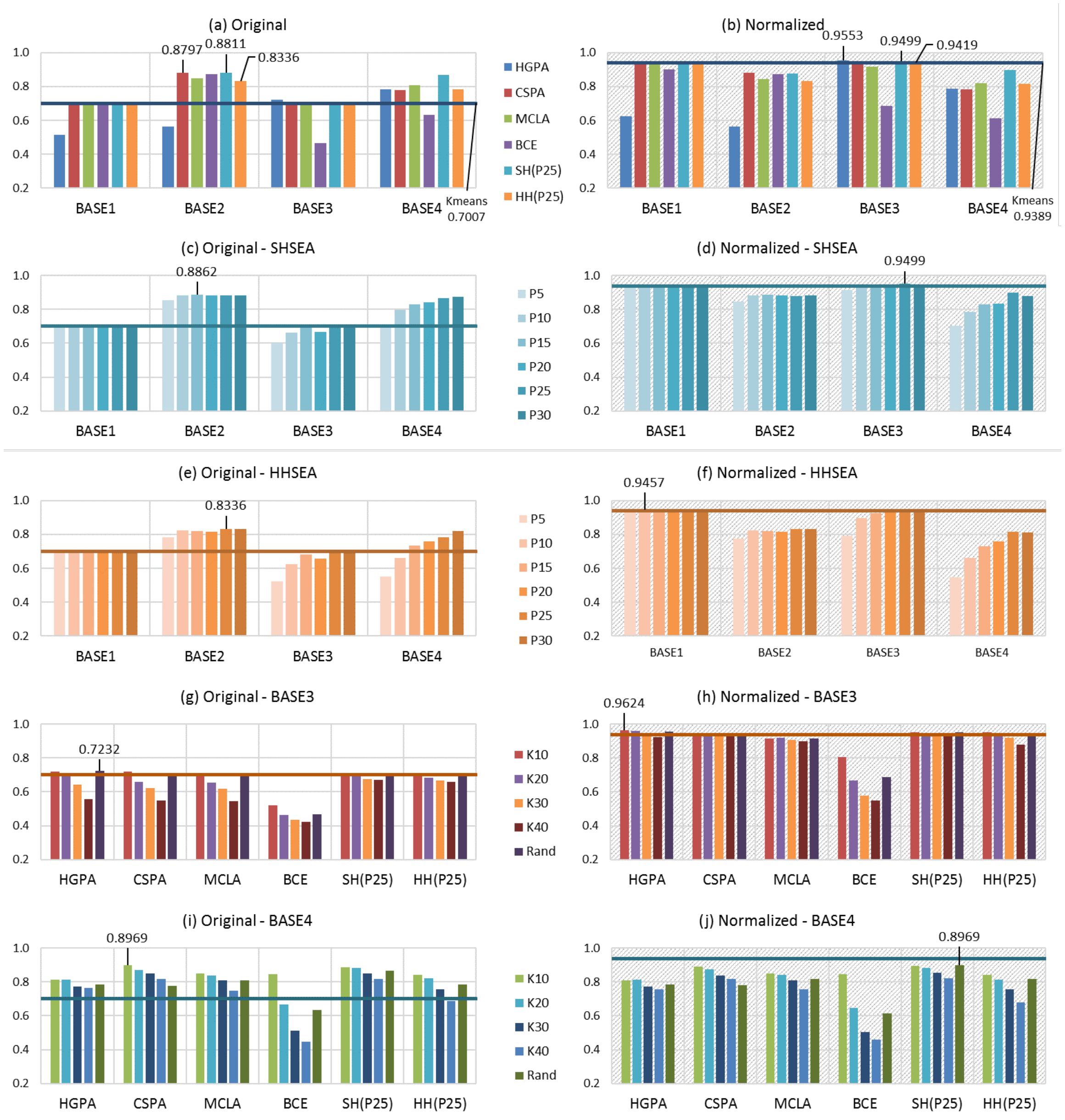


Figure 5.5: Original vs. Normalized Wine

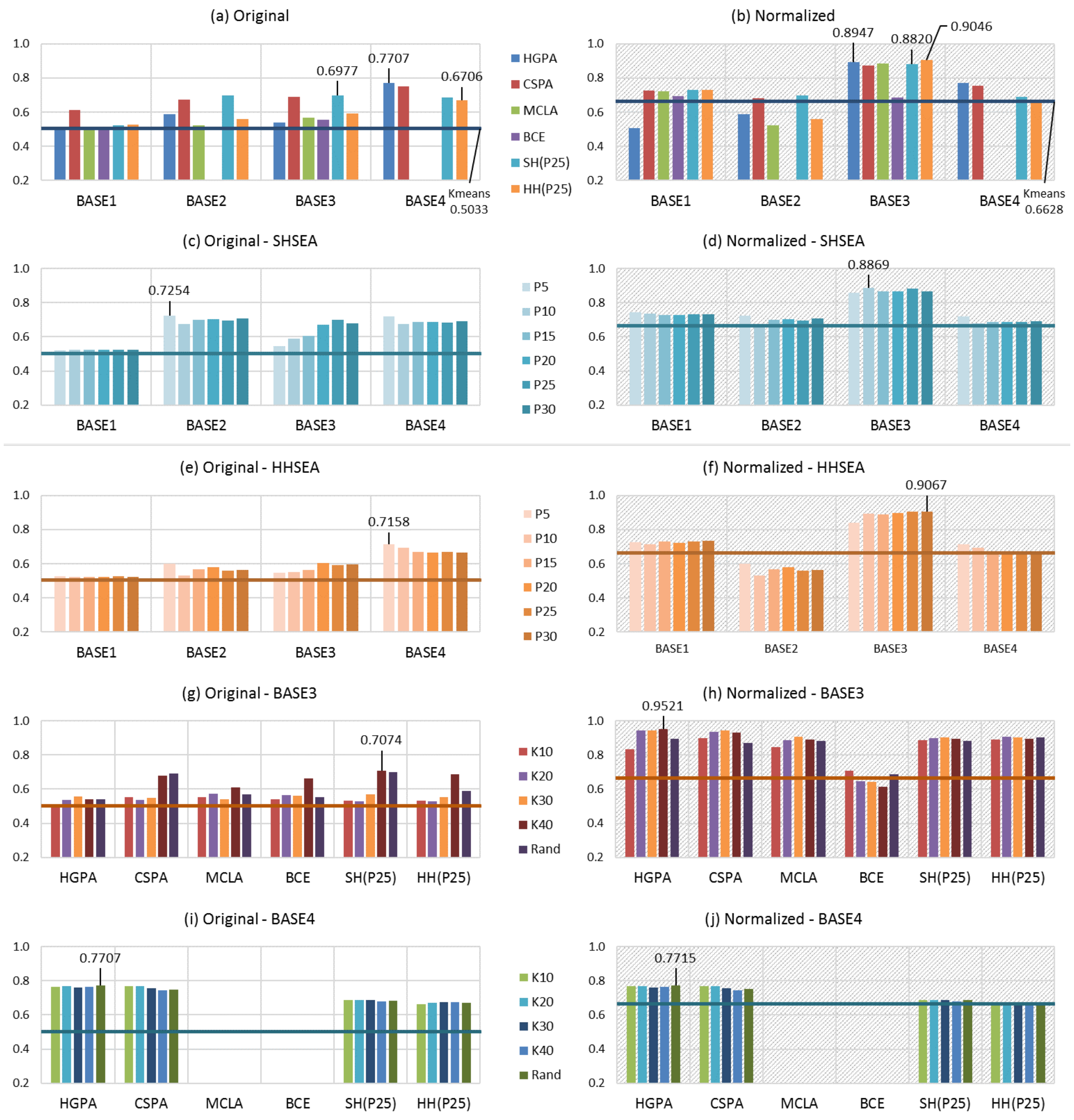


Figure 5.6: Original vs. Normalized DWALabSet1

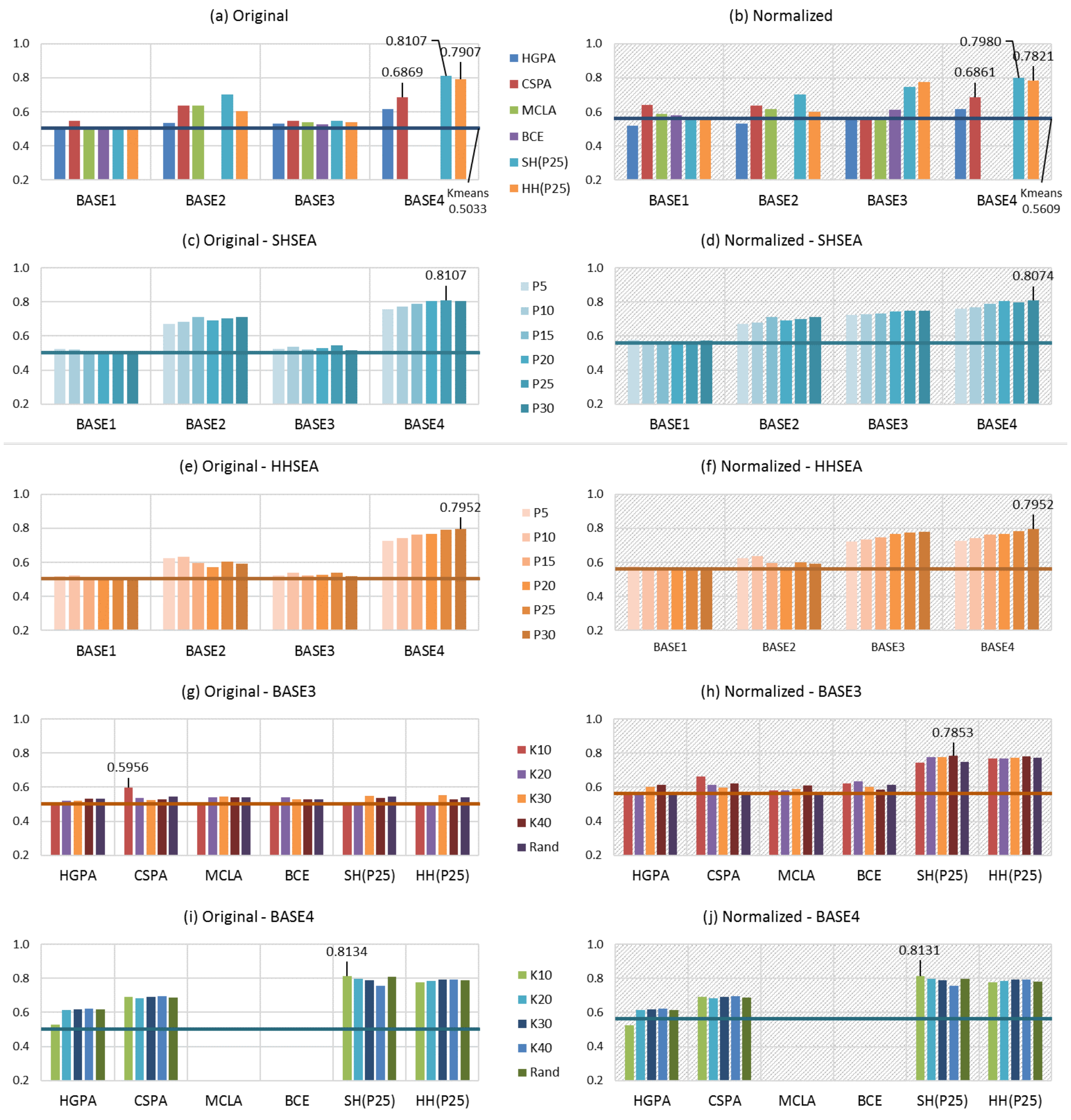


Figure 5.7: Original vs. Normalized DWALabSet2

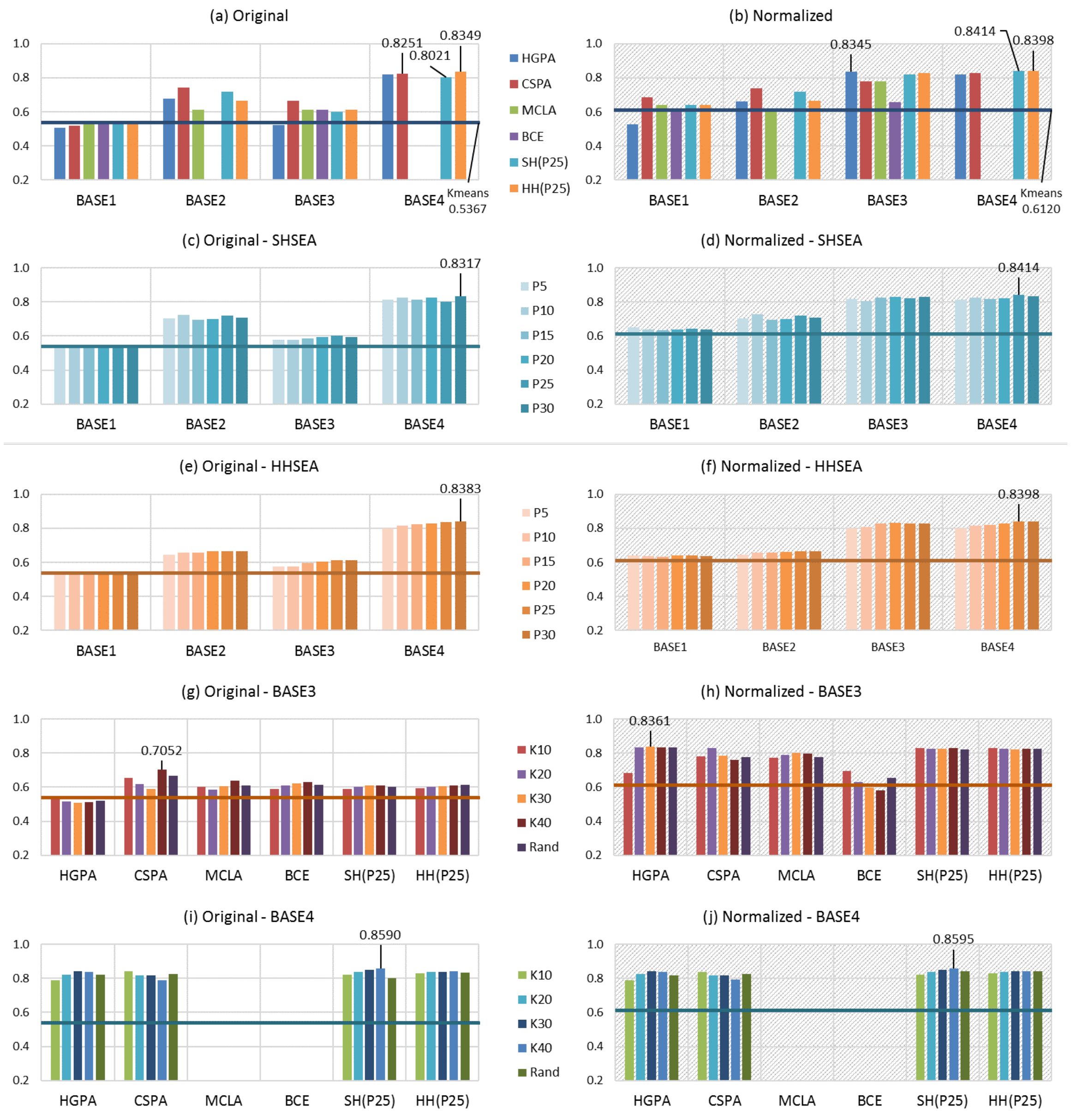


Figure 5.8: Original vs. Normalized DWALabSet3

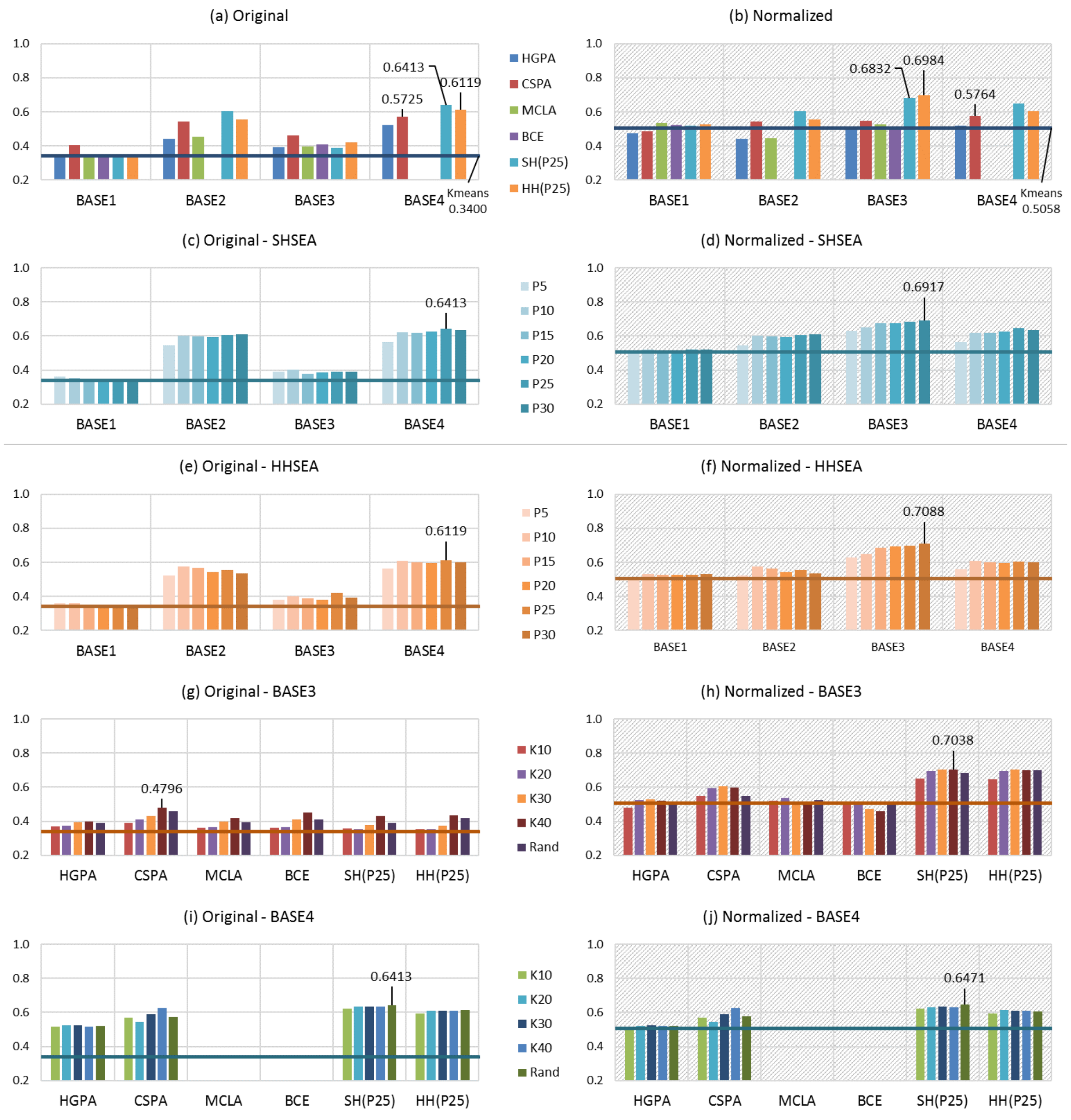


Figure 5.9: Original vs. Normalized DWALabSet4

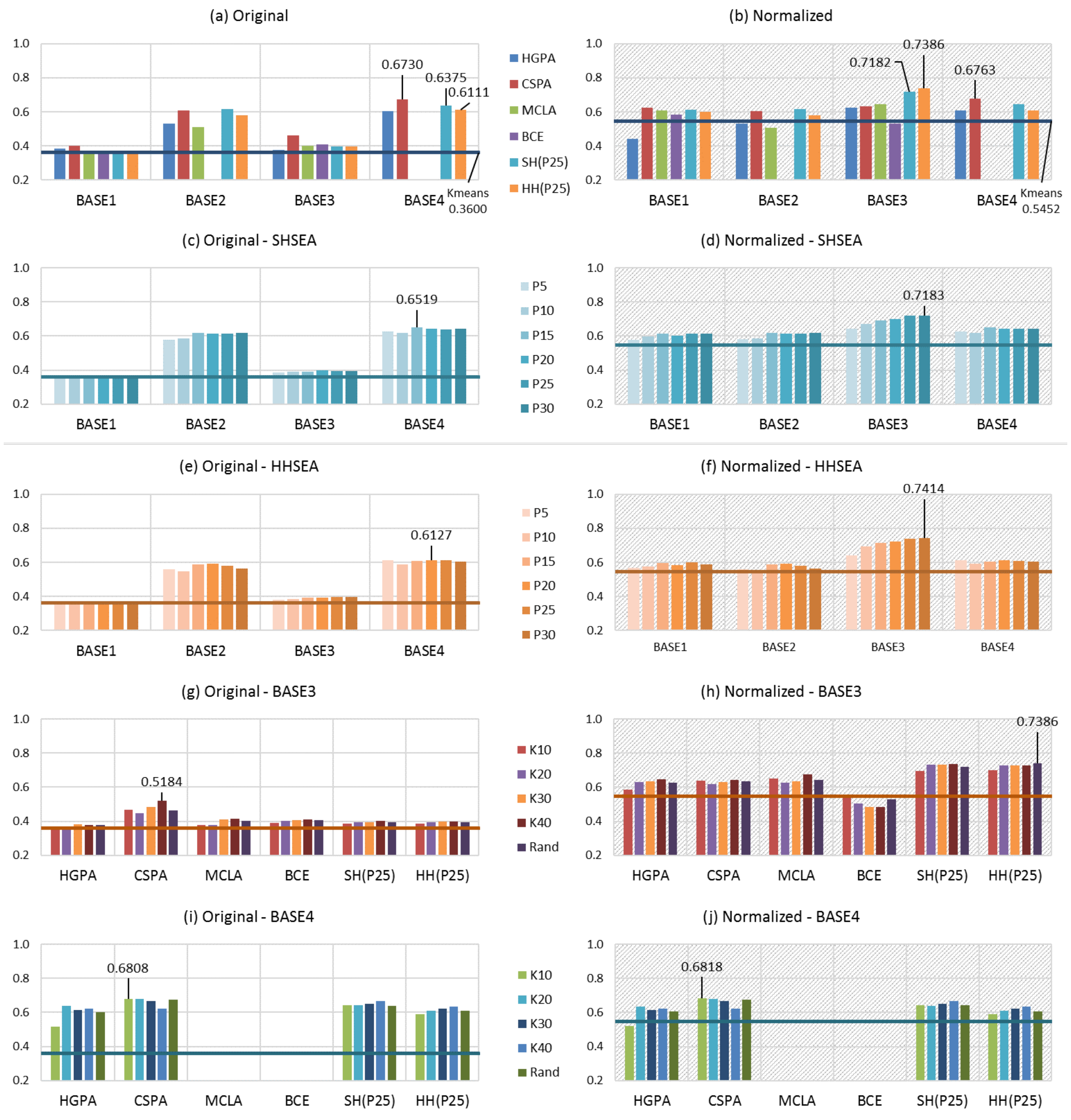


Figure 5.10: Original vs. Normalized DWALabSet5

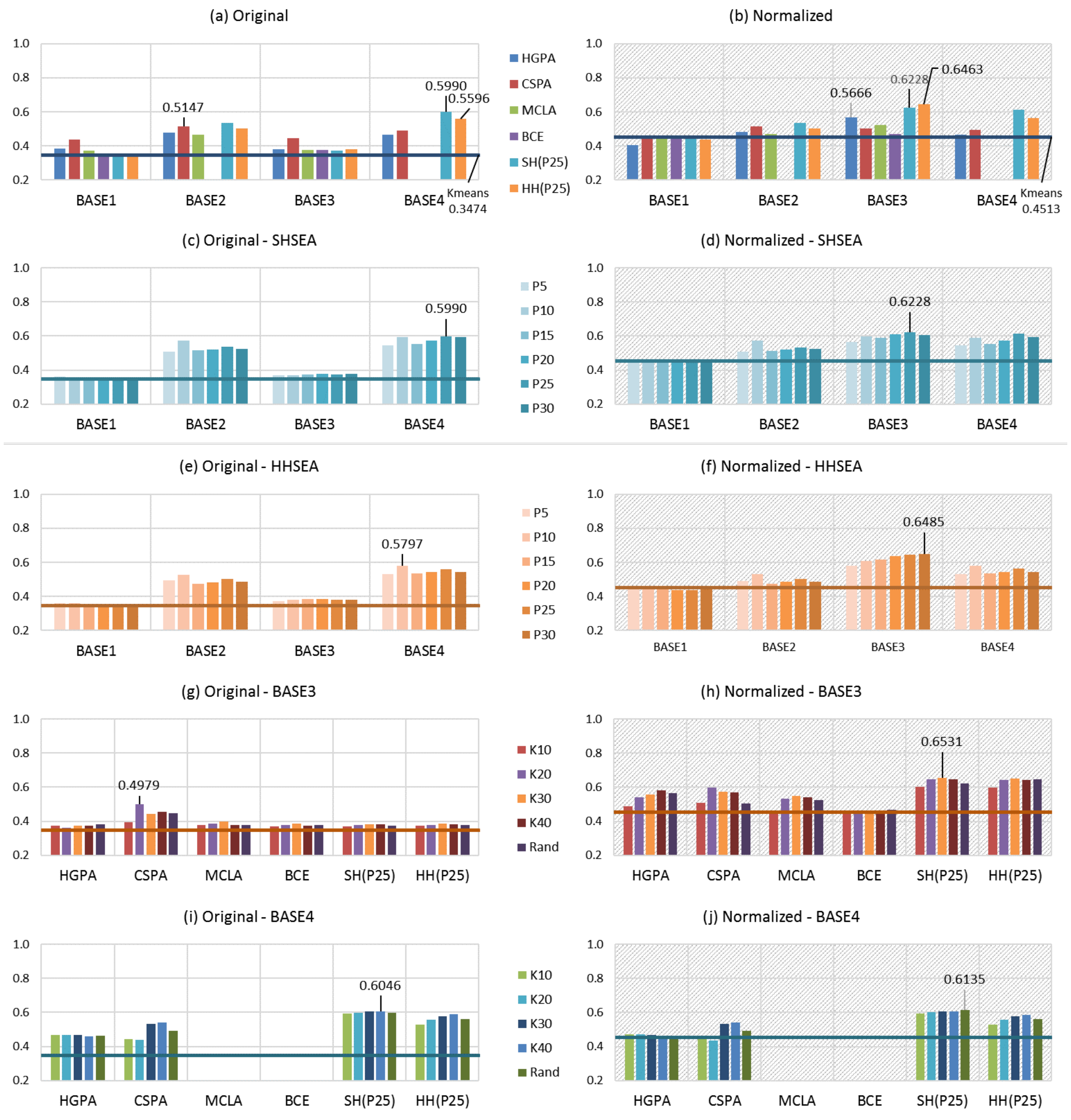


Figure 5.11: Original vs. Normalized DWALabSet6

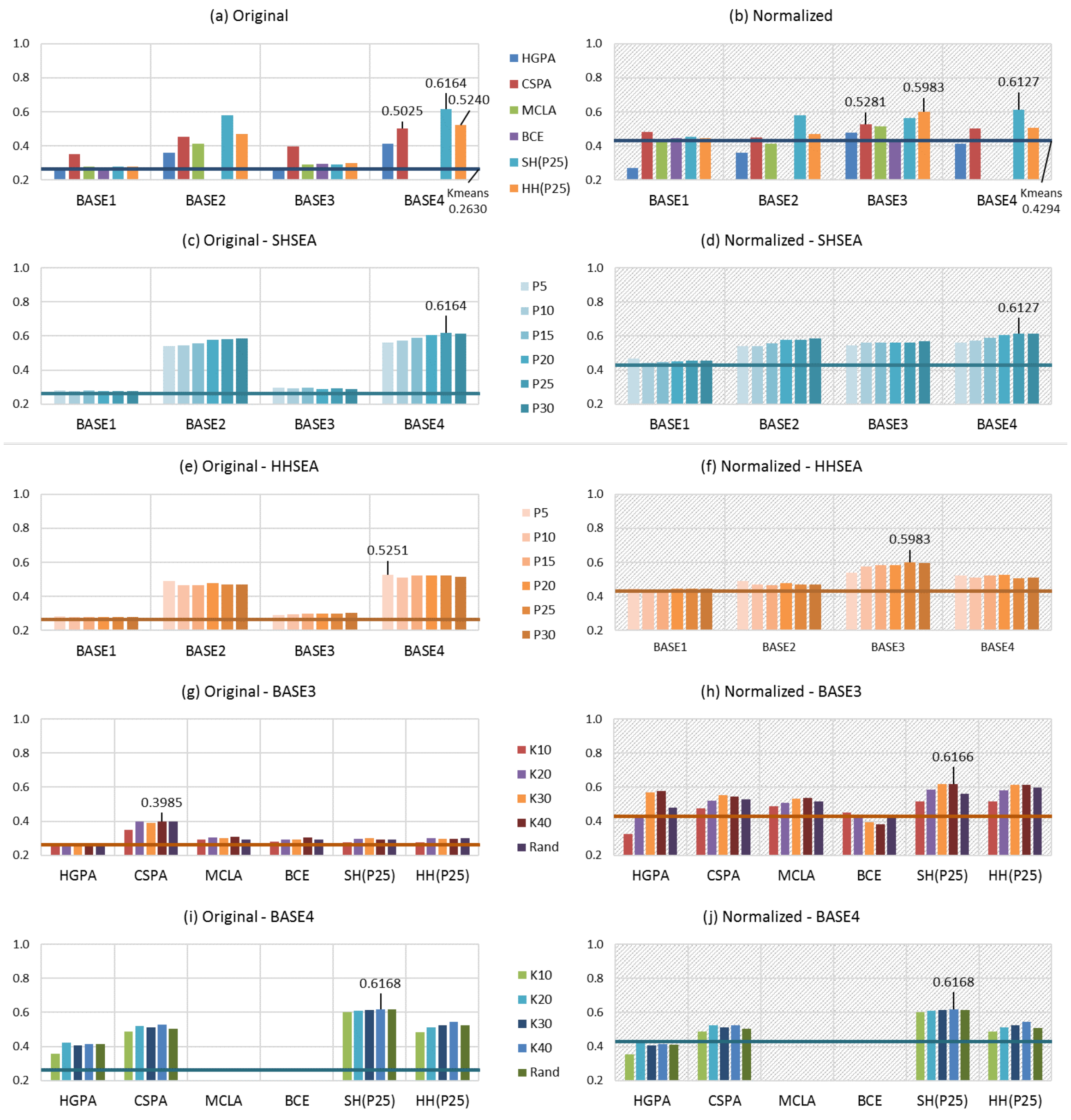


Figure 5.12: Original vs. Normalized DWALabSet7

Table 5.4: Comparison of unsupervised mode and semi-supervised mode using original data sets

Data Sets	Kmeans	Unsupervised				SHSEA			HHSEA		
	MP	Winner	Base Set	MP	Improved %	Base Set	MP	Improved %	Base Set	MP	Improved %
Ionosphere	0.7111	BCE	4	0.7670	5.6%	4	0.8886	17.7%	**3**	**0.8899**	**17.9%**
Pima	0.6532	MCLA	2	0.7039	5.1%	**2**	**0.7325**	**7.9%**	3	0.7070	5.4%
Balance	0.5498	MCLA	1	0.5945	4.5%	1	0.6922	14.2%	**1**	**0.7262**	**17.6%**
Wine	0.7007	CSPA	2	0.8797	17.9%	**2**	**0.8811**	**18.0%**	2	0.8336	13.3%
DWALabSet1	0.5033	**HGPA**	**4**	**0.7707**	**26.7%**	3	0.6977	19.4%	4	0.6706	16.7%
DWALabSet2	0.5033	CSPA	4	0.6869	18.4%	**4**	**0.8107**	**30.7%**	4	0.7907	28.7%
DWALabSet3	0.5367	CSPA	4	0.8251	28.8%	4	0.8021	26.5%	**4**	**0.8349**	**29.8%**
DWALabSet4	0.3400	CSPA	4	0.5725	23.3%	**4**	**0.6413**	**30.1%**	4	0.6119	27.2%
DWALabSet5	0.3600	CSPA	4	0.6730	31.3%	4	0.6375	27.8%	**4**	**0.6730**	**31.3%**
DWALabSet6	0.3474	CSPA	2	0.5147	16.7%	**4**	**0.5990**	**25.2%**	4	0.5596	21.2%
DWALabSet7	0.2630	CSPA	4	0.5025	24.0%	**4**	**0.6164**	**35.3%**	4	0.5240	26.1%

methods is labelled in each chart. Compared to the performance of K-means (Table 5.3), the micro-precision of SHSEA or HHSEA increases dramatically when P is relatively small. It becomes steady and sometimes starts to decrease as P increases. Therefore, for the purpose of improving the performance of semi-supervised ensemble algorithms it is not worth to make effort on labelling more data points. It is due to the facts that more reference points do not guarantee the improvement and obtaining additional labels is time-consuming and expensive.

Recall that the number of clusters in the j-th base clustering $K^{(j)}$ is randomly generated in the base clustering generator $\mathbf{\Phi_3}$ and $\mathbf{\Phi_4}$. To study the effect of randomized $K^{(j)}$ on the clustering ensemble methods, we repeat the experiments by setting the number of clusters in each base clustering the same and varying the value of $K^{(j)}$. The micro-precision of our proposed system using different values of $K^{(j)}$ is illustrated by sub-figures (g) and (i) of Figures 5.2 to 5.12. The highest MP using BASE3 and BASE4 are labelled in the corresponding chart. Among these data sets, the highest MP occurs at different $K^{(j)}$. The performance of the proposed system using randomized $K^{(j)}$ is either the best of all tested valuses of $K^{(j)}$ or it is very closed to the best. Due to the fact that we lack the knowledge on how to select the optimal $K^{(j)}$, we use randomized $K^{(j)}$ in the following experiments to avoid the selection of $K^{(j)}$ for each data set.

5.2.2 Normalized Data Sets

In Section 4.2 we reviewed the Min-max and Z-score normalization methods that are commonly used in clustering. To study the effect of normalization on clustering

Table 5.5: Micro-precision of K-means using normalized data sets: Norm1, Norm2 and Norm3 refer to normalizing data using the Min-max method into ranges of [0,1], [0,10] and [0, 100] respectively; Norm4 refers to normalizing data using the Z-score method

Data Sets	Original	Norm1 [0, 1]	Norm2 [0, 10]	Norm3 [0, 100]	Norm4 Z-score
Ionosphere	0.7111	0.7111	0.7111	0.7111	0.7111
Pima	0.6532	0.6627	0.6627	0.6628	0.6639
Balance	0.5498	0.5520	0.5510	0.5507	0.5470
Wine	0.7007	0.9389	0.9408	0.9361	0.9467
DWALabSet1	0.5033	0.6628	0.6720	0.6646	0.6082
DWALabSet2	0.5033	0.5609	0.5629	0.5605	0.5516
DWALabSet3	0.5367	0.6120	0.6171	0.6159	0.6132
DWALabSet4	0.3400	0.5058	0.5008	0.5014	0.4759
DWALabSet5	0.3600	0.5452	0.5535	0.5395	0.4488
DWALabSet6	0.3474	0.4513	0.4524	0.4510	0.4277
DWALabSet7	0.2630	0.4294	0.4299	0.4324	0.3955

algorithms, we first normalized each data set using these methods and then applied K-means to the normalized data sets. The micro-precision of K-means using normalized data is listed in Table 5.5, where Norm1, Norm2 and Norm3 refer to normalizing data using the Min-max method into ranges of [0,1], [0,10] and [0, 100] respectively, and Norm4 refers to normalizing data using the Z-score method. There is no significant difference on the performance of K-means when each data set is normalized into different ranges by the Min-max method. The micro-precision of K-means using data sets normalized by the Z-score method is slightly lower than those normalized by the Min-max method. Therefore, in the following experiments when normalization is necessary we normalize data sets into [0, 1] using the Min-max method, except otherwise stated.

The micro-precision of K-means using all normalized features and normalized features individually is shown in Table 5.6. The performance of K-means using all features has been improved significantly by normalization except the first three data

Table 5.6: Micro-precision of K-means using all features and single feature of normalized data

Data Sets (Normailzed)	Kmeans		
	All Features	Single Feature	
		Max	Min
Ionosphere	0.7111	0.7804	0.5148
Pima	0.6627	0.7107	0.5091
Balance	0.5520	0.5192	0.5086
Wine	0.9389	0.7883	0.3740
DWALabSet1	0.6628	0.7920	0.5000
DWALabSet2	0.5609	0.7233	0.5000
DWALabSet3	0.6120	0.7933	0.5000
DWALabSet4	0.5058	0.5644	0.3333
DWALabSet5	0.5452	0.5578	0.3388
DWALabSet6	0.4513	0.5440	0.3355
DWALabSet7	0.4294	0.4433	0.2583

sets, as compared to Table 5.3. As discussed earlier the performance of distance-based clustering algorithms may be affected when data sets to be clustered contain features measured in diverse scales. By investigating features of each data set, we notice that data set“Wine” from UCI and all data sets from DWALab contain features measured in quite different ranges. Moreover, the performance of K-means using normalized features individually is similar to the performance of K-means using original features individually. This result is expected since the similarity measure for a single feature is based on 1-dimensional distance calculation and is invariant to the feature scales.

To study the effect of normalization on clustering ensemble methods, we repeat the experiments previously described in Section 5.2.1 using normalized data sets. The micro-precision of the proposed system is illustrated by sub-figure (b) of Figures 5.2 to 5.12. As compared to the system performance using the original data sets, there is no siginificant difference on the performance when the input of the sytem is switched to normalized “Ionosphere”, “Pima” and “Balance”. It is expected since normalization is not able to boost the performance of K-means on these data sets. For the other data sets, the system performance using BASE1 and BASE3 has been improved by normalization, while the system performance using BASE2 and BASE4 stays close to the system performance using the corresponding sets of base clusterings obtained by clustering original data sets. It is also expected since normalization does not affect the performance of K-means using single feature.

For each data set, the winner of unsupervised methods, its corresponding winning set of base clusterings, and the winning sets of SHSEA and HHSEA are summarized in Table 5.7. The winner of unsupervised methods is still data dependent. The performance of semi-supervised mode is better than the unsupervised mode except the "Wine" data set. The performance of SHSEA is very close to the performance of HHSEA using normalized data.

5.2.3 Extracted Features

To study the effect of feature extraction on clustering algorithms, we perform principal component analysis (PCA) on the original data sets and apply clustering algorithm afterwards. We start the experiment by retaining 80% of total data variations and increase the percentage to 95%. The micro-precision of K-means and the number of principal components that is required to retain the corresponding level of data variations are given in Table 5.8. The performance of K-means using the extracted features stays close to the performance of K-means using all original features, i.e., there is no significant improvement on the performance of K-means by feature extraction. However, the computational complexity has been reduced significantly since the number of principal components (features) decreases dramatically.

We repeat the experiments using normalized data sets. The micro-precision of

Table 5.7: Comparison of unsupervised mode and semi-supervised mode using normalized data sets

Data Sets	Kmeans	Unsupervised				SHSEA			HHSEA		
(Normalized)	MP	Winner	Base Set	MP	Improved %	Base Set	MP	Improved %	Base Set	MP	Improved %
Ionosphere	0.7111	BCE	4	0.7537	4.3%	**4**	**0.8859**	**17.5%**	3	0.8748	16.4%
Pima	0.6623	MCLA	2	0.7030	4.1%	**2**	**0.7319**	**7.0%**	3	0.7039	4.2%
Balance	0.5498	MCLA	1	0.5968	4.7%	4	0.6910	14.1%	**3**	**0.7111**	**16.1%**
Wine	0.9357	HGPA	**3**	**0.9553**	**2.0%**	4	0.9449	0.9%	3	0.9419	0.6%
DWALabSet1	0.6686	HGPA	3	0.8947	22.6%	3	0.8820	21.3%	**3**	**0.9046**	**23.6%**
DWALabSet2	0.5612	CSPA	4	0.6861	12.5%	**4**	**0.7980**	**23.7%**	4	0.7821	22.1%
DWALabSet3	0.6126	HGPA	3	0.8345	22.2%	**4**	**0.8414**	**22.9%**	4	0.8398	22.7%
DWALabSet4	0.4991	CSPA	4	0.5764	7.7%	3	0.6832	18.4%	**3**	**0.6984**	**19.9%**
DWALabSet5	0.5596	CSPA	4	0.6763	11.7%	3	0.7182	15.9%	**3**	**0.7386**	**17.9%**
DWALabSet6	0.4521	HGPA	3	0.5666	11.5%	3	0.6228	17.1%	**3**	**0.6463**	**19.4%**
DWALabSet7	0.4259	CSPA	4	0.5039	7.8%	**4**	**0.6127**	**18.7%**	3	0.5983	17.2%

K-means and the corresponding number of principal components is listed in Table 5.9. The performance of K-means using the extracted normalized features stays close to the performance of K-means using all normalized features. The more data variation is retained the more principal components are required. However, there is no significant difference on the performance of K-means when different percentages of data variation are retained. Therefore, the major advantage of performing PCA before clustering is not to increase the accuracy of the clustering results, but rather to reduce the data dimensionality. In the following experiments, extracted features (or PCA data) refer to the principal components obtained by retaining 80% of data variation except otherwise stated.

The performance of our proposed system using extracted original features is illustrated by sub-figure (c) of Figures 5.13 to 5.23, while the performance using extracted normalized features is illustrated by sub-figure (d). For easy comparison purpose, the system performance using the original and normalized data sets (all features) is displayed again in these figures. Since the micro-precision of K-means was not improved by performing PCA, the micro-precision of our proposed system does not change very much as expected. The winner of the unsupervised methods, its corresponding winning set of base clusterings, and the winning set of SHSEA and HHSEA are summarized in Table 5.10. The winner of unsupervised methods and the winning set of base clusterings are still data dependent. In these experiments semi-supervised mode performs better than unsupervised mode on all data sets. The performance of HHSEA is slightly better than the performance of SHSEA on the majority of normalized data sets. The overall winning set of base clusterings is BASE3.

Table 5.8: Micro-precision of K-means using extracted features (of original data) obtained by retaining different percentages of data variation: column "No. of PCs" records the number of principal components used to retain the corresponding percentage of data variation

Data Sets	All Features		Extracted Features (PCA)							
			80% Variations		85% Variations		90% Variations		95% Variations	
	No. of Features	MP	No. of PCs	MP	No. of PCs	MP	No. of PCs	MP	No. of PCs	MP
Ionosphere	34	0.7111	11	0.7111	14	0.7111	18	0.7111	23	0.7111
Pima	8	0.6532	1	0.6548	1	0.6548	2	0.6528	2	0.6531
Balance	4	0.5498	4	0.5501	4	0.5520	4	0.5522	4	0.5542
Wine	13	0.7007	1	0.7007	1	0.7007	1	0.7007	1	0.7007
DWALabSet1	705	0.5033	1	0.5033	1	0.5033	1	0.5033	1	0.5033
DWALabSet2	705	0.5033	1	0.5033	2	0.5033	2	0.5033	2	0.5033
DWALabSet3	705	0.5367	1	0.5367	2	0.5367	2	0.5367	2	0.5367
DWALabSet4	705	0.3400	1	0.3422	2	0.3400	2	0.3400	2	0.3400
DWALabSet5	705	0.3600	1	0.3600	1	0.3600	2	0.3600	2	0.3600
DWALabSet6	705	0.3474	1	0.3489	1	0.3489	1	0.3489	2	0.3469
DWALabSet7	705	0.2630	1	0.2650	1	0.2650	2	0.2628	2	0.2628

Table 5.9: Micro-precision of K-means using extracted features (of normalized data) obtained by retaining different percentages of data variation: column "No. of PCs" records the number of principal components used to retain the corresponding percentage of data variation

Data Sets Normalized	All Features		Extracted Features (PCA)							
			80% Variations		85% Variations		90% Variations		95% Variations	
	No. of Features	MPs	No. of PCs	Mps	No. of PCs	MPs	No. of PCs	MPs	No. of PCs	MPs
Ionosphere	34	0.7111	11	0.7111	14	0.7111	18	0.7111	23	0.7111
Pima	8	0.6627	5	0.6684	6	0.6648	6	0.6648	7	0.6643
Balance	4	0.5520	4	0.5511	4	0.5516	4	0.5489	4	0.5545
Segmentation	13	0.9389	5	0.9362	6	0.9356	8	0.9355	10	0.9390
DWALabSet1	705	0.6628	25	0.6682	34	0.6850	51	0.6768	84	0.6547
DWALabSet2	705	0.5609	26	0.5590	37	0.5510	54	0.5526	88	0.5552
DWALabSet3	705	0.6120	21	0.6117	30	0.6010	47	0.6071	80	0.6089
DWALabSet4	705	0.5058	25	0.5069	37	0.4982	56	0.5020	95	0.4972
DWALabSet5	705	0.5452	23	0.5475	33	0.5550	52	0.5408	91	0.5550
DWALabSet6	705	0.4513	24	0.4507	35	0.4511	54	0.4499	94	0.4543
DWALabSet7	705	0.4294	24	0.4291	36	0.4289	56	0.4289	98	0.4284

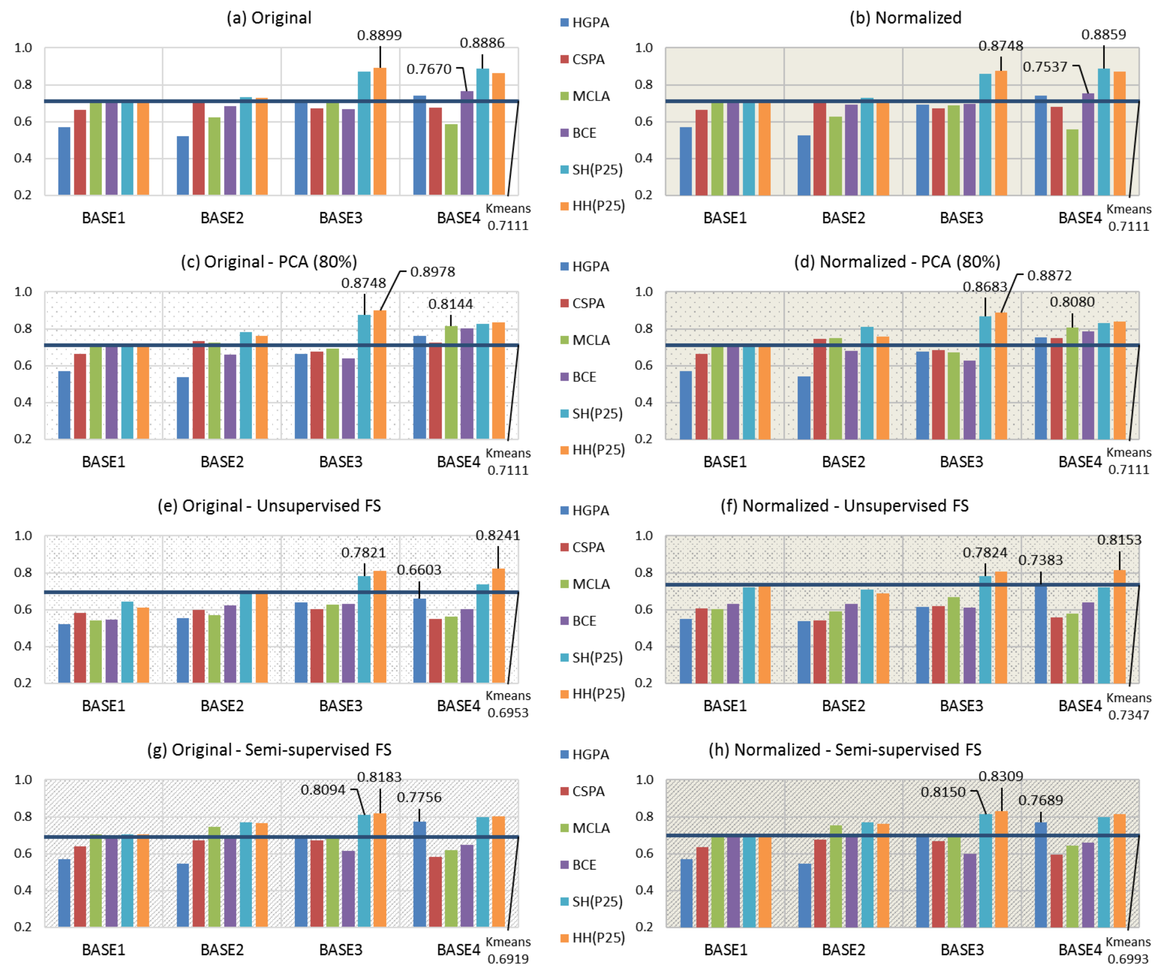


Figure 5.13: Feature extracted vs. Feature selected Ionosphere

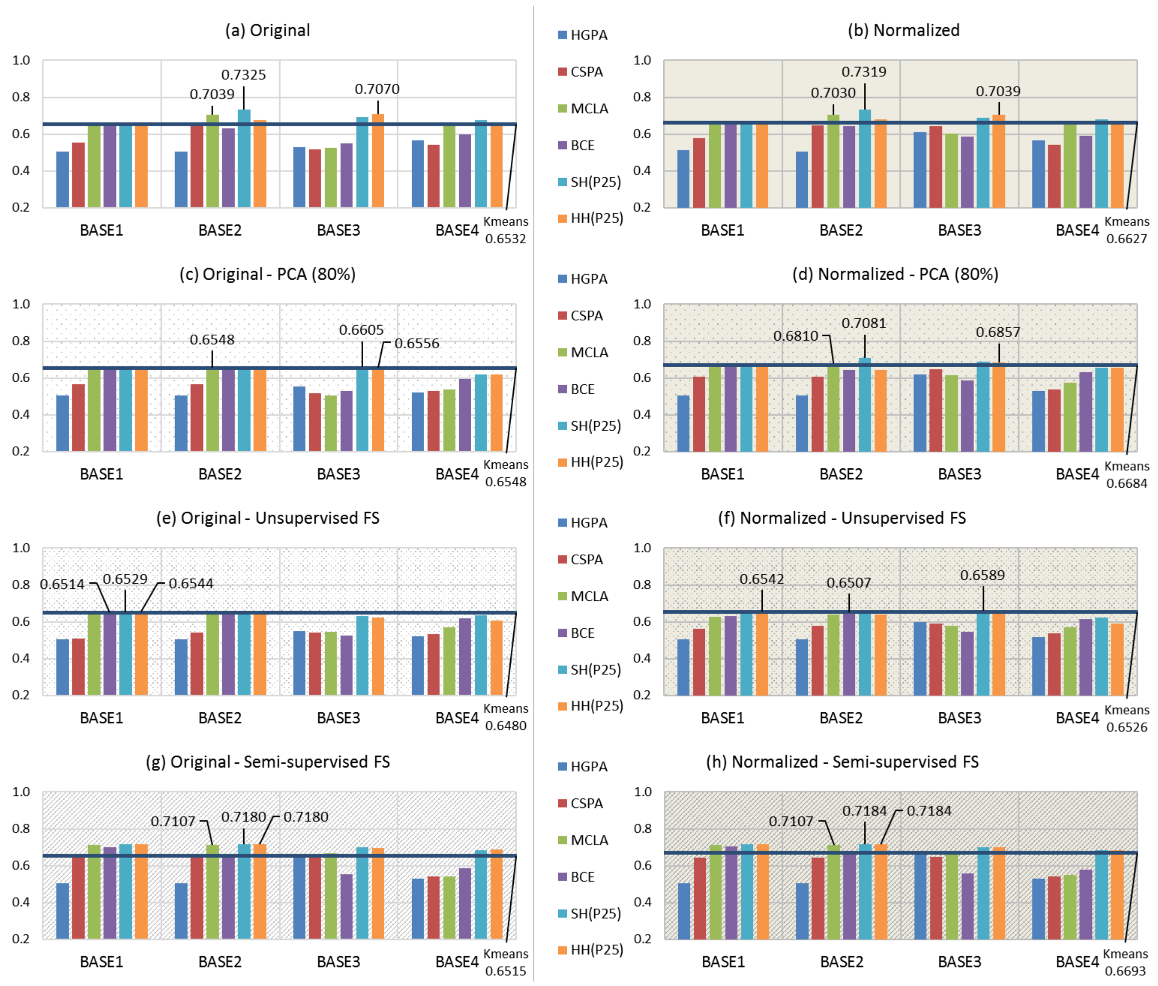


Figure 5.14: Feature extracted vs. Feature selected Pima

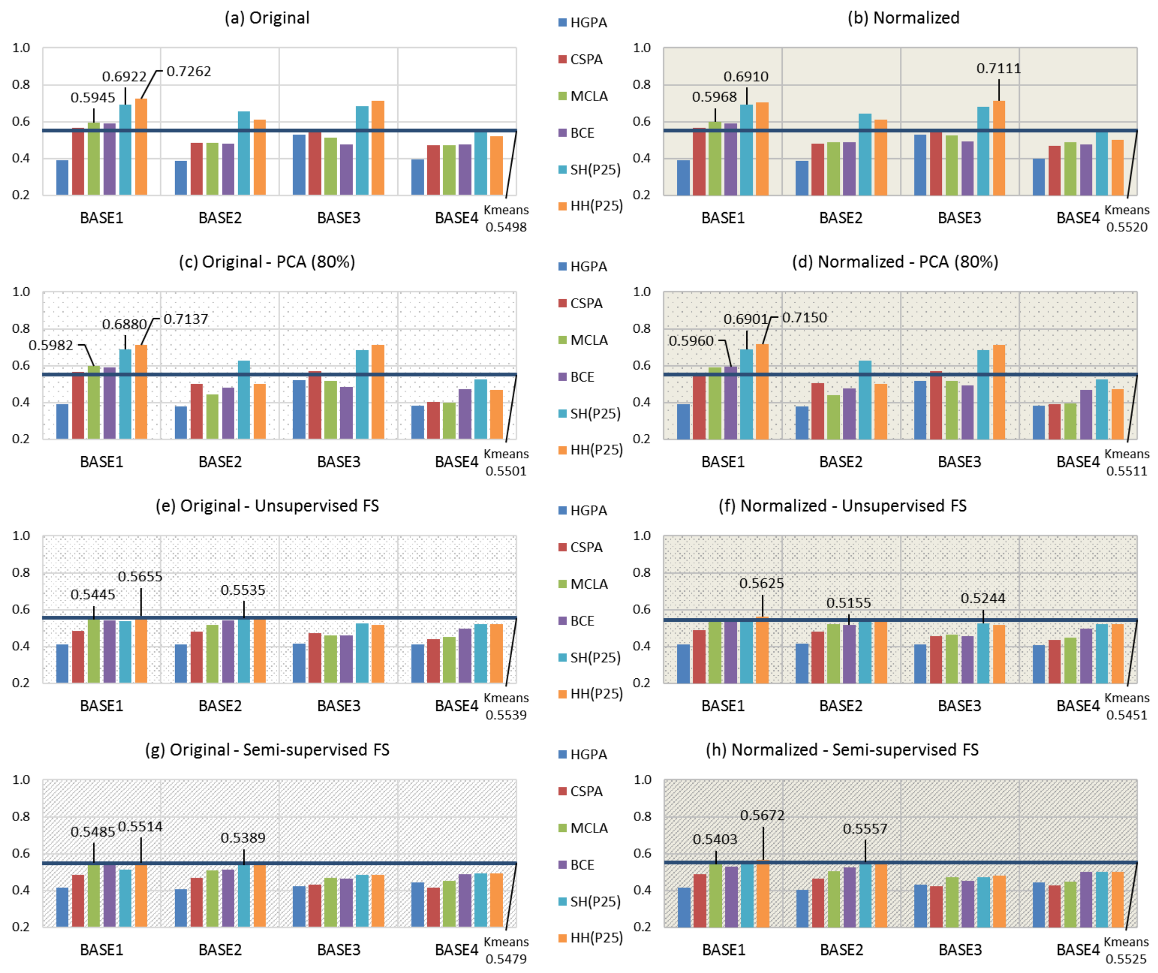


Figure 5.15: Feature extracted vs. Feature selected Balance

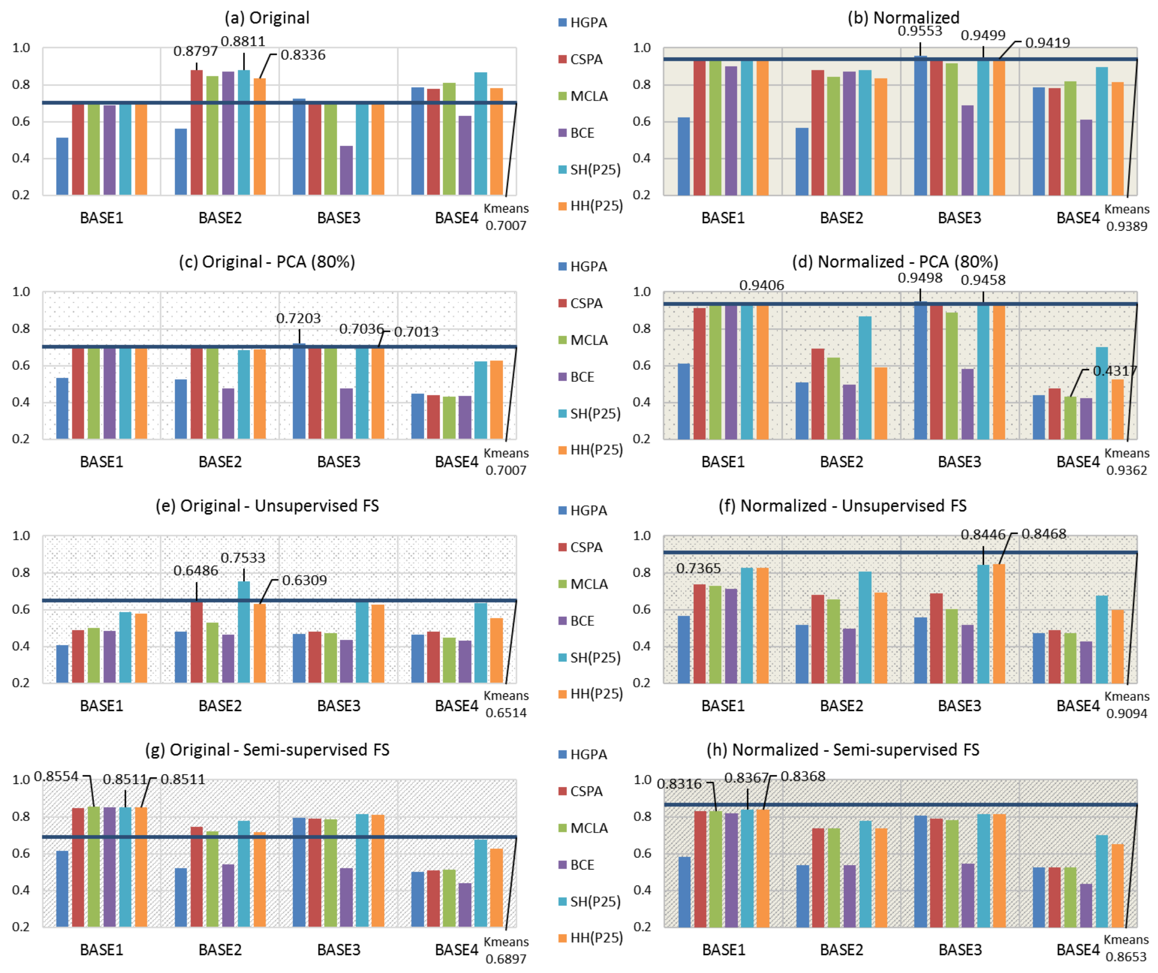


Figure 5.16: Feature extracted vs. Feature selected Wine

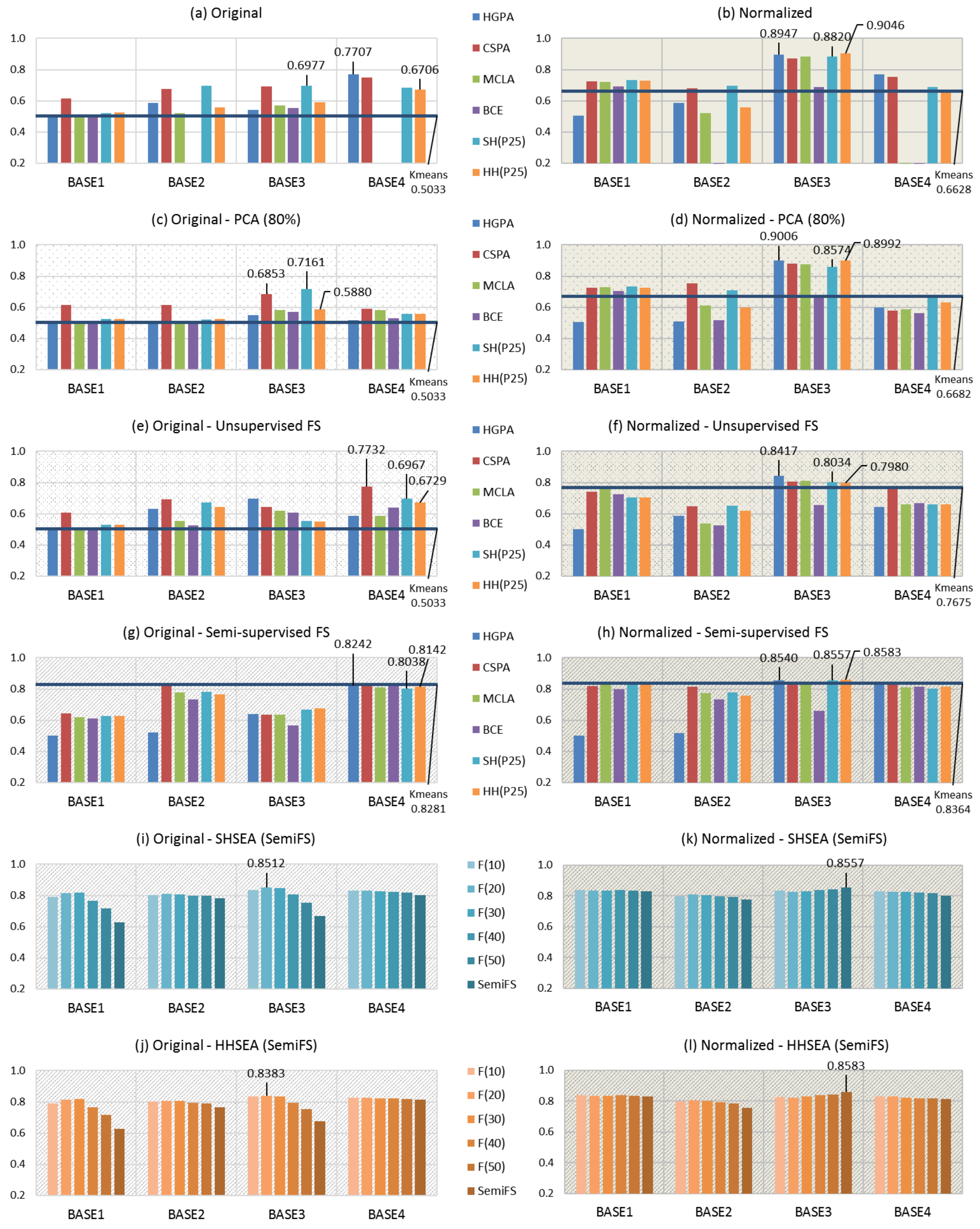


Figure 5.17: Feature extracted vs. Feature selected DWALabSet1

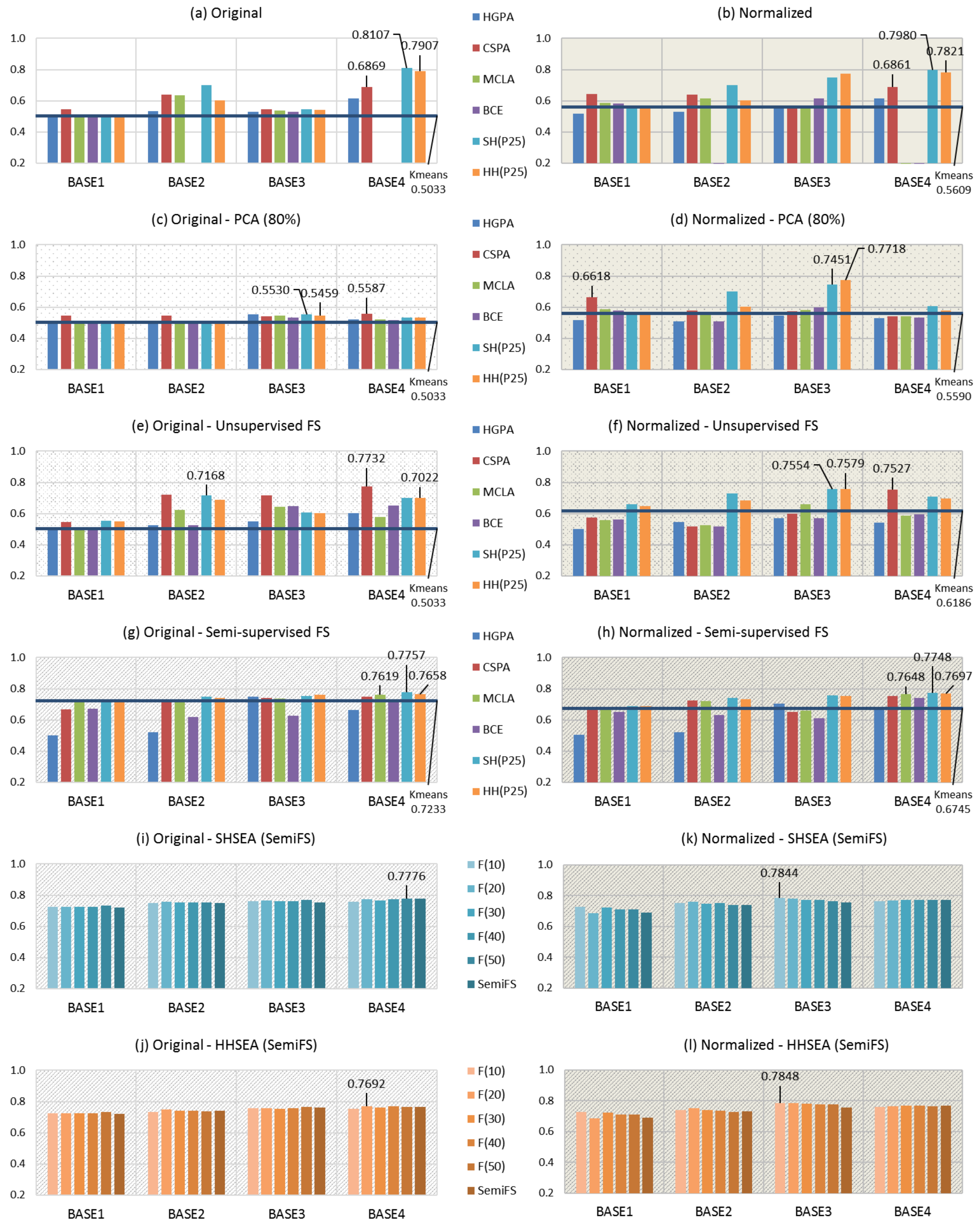


Figure 5.18: Feature extracted vs. Feature selected DWALabSet2

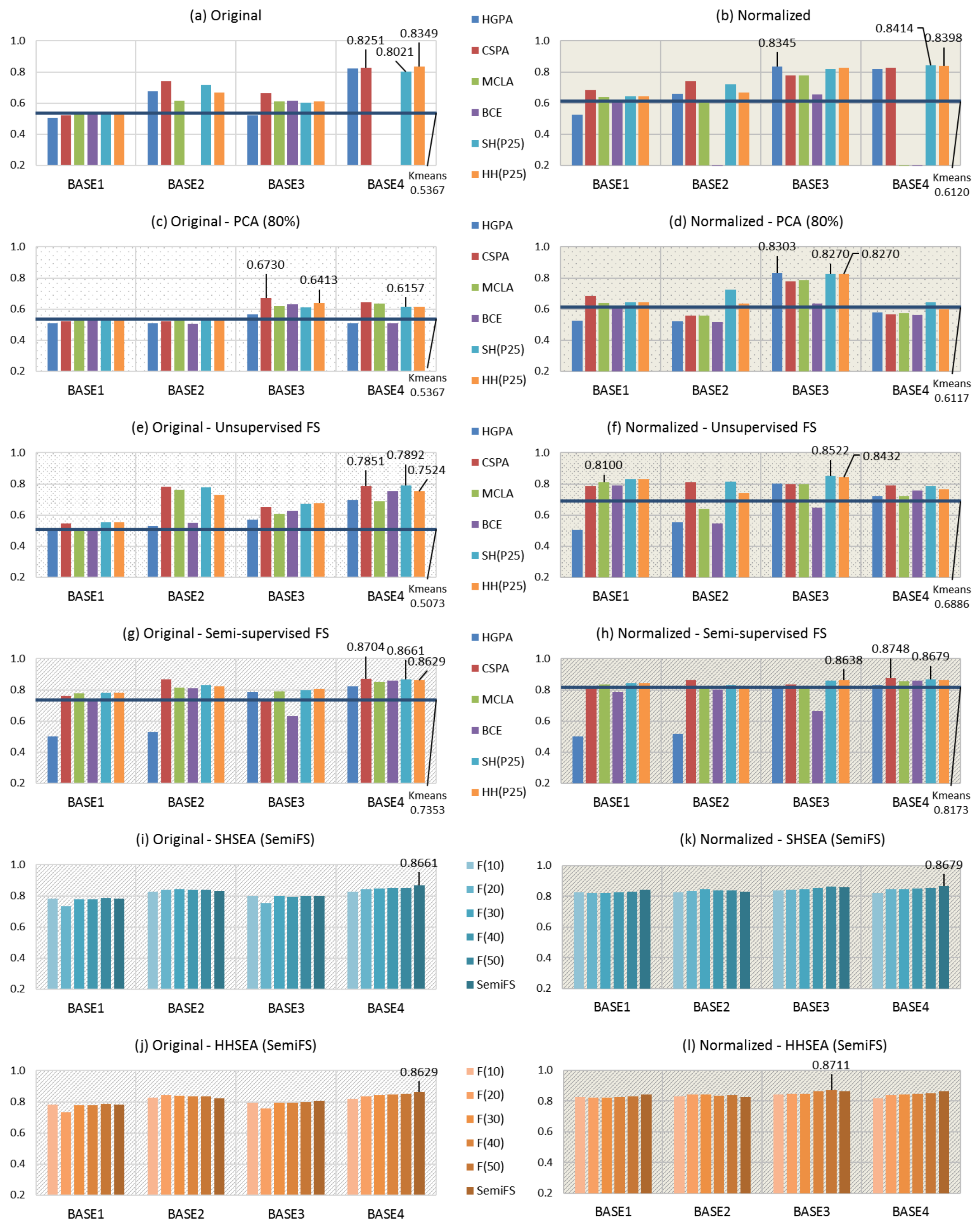


Figure 5.19: Feature extracted vs. Feature selected DWALabSet3

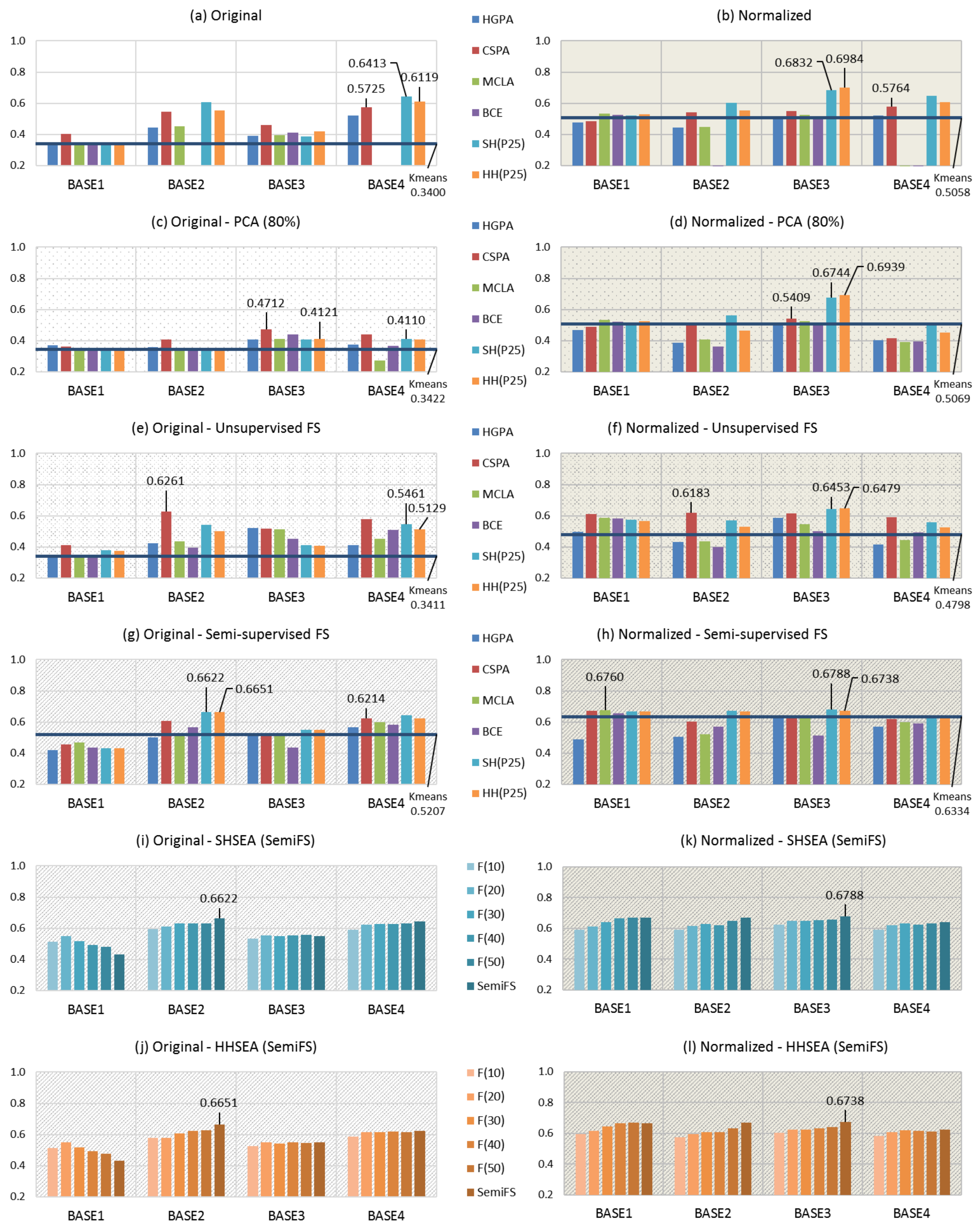


Figure 5.20: Feature extracted vs. Feature selected DWALabSet4

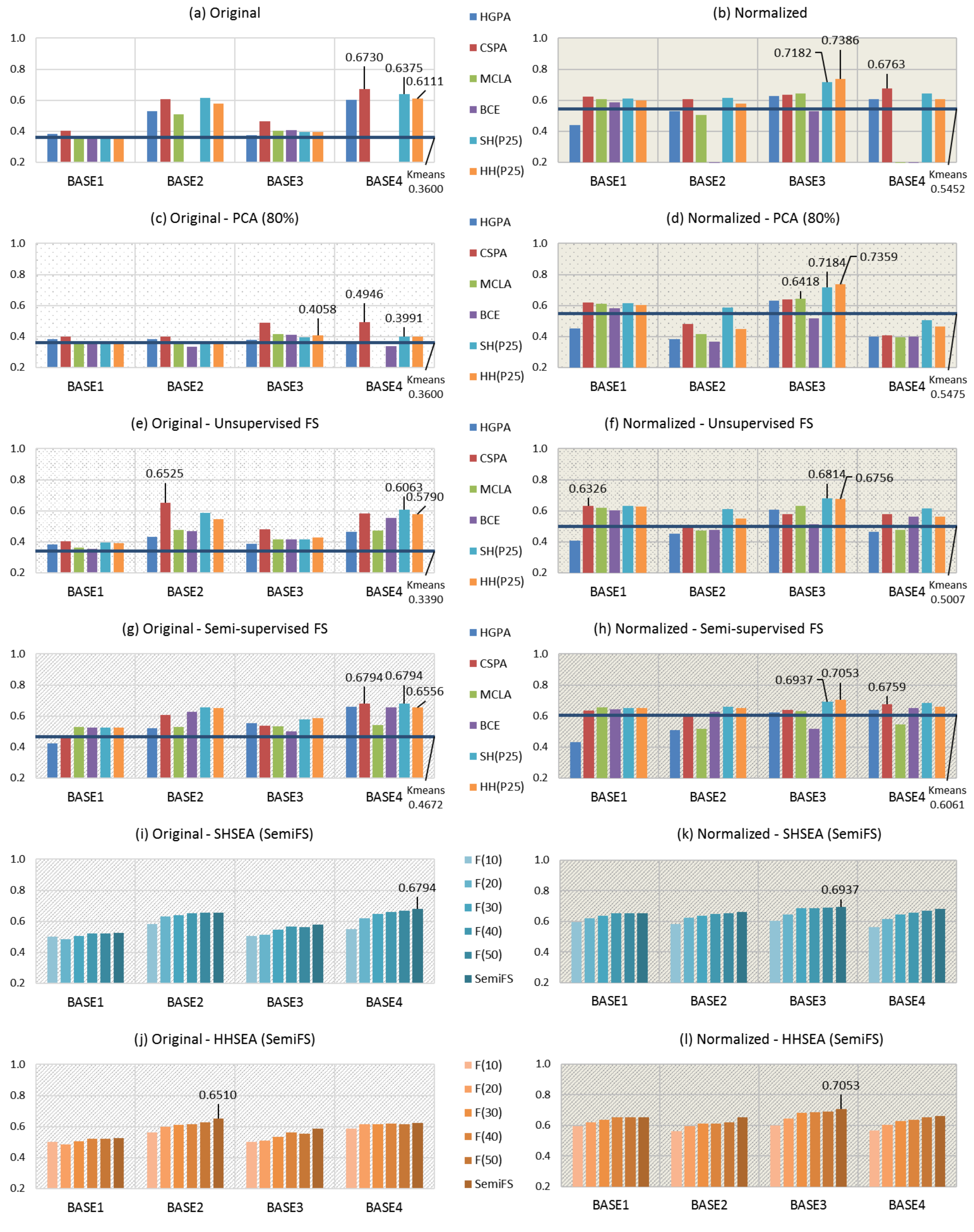


Figure 5.21: Feature extracted vs. Feature selected DWALabSet5

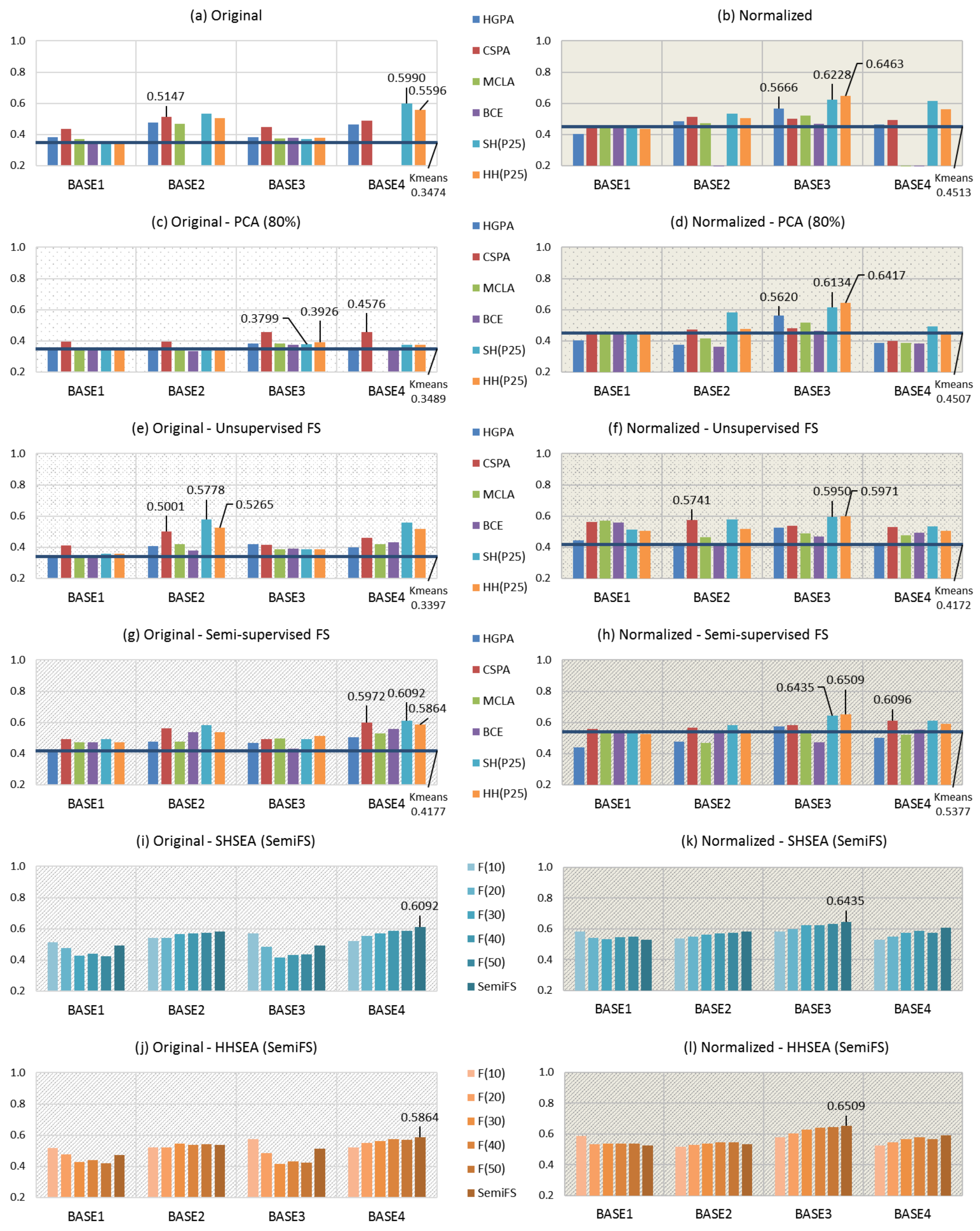


Figure 5.22: Feature extracted vs. Feature selected DWALabSet6

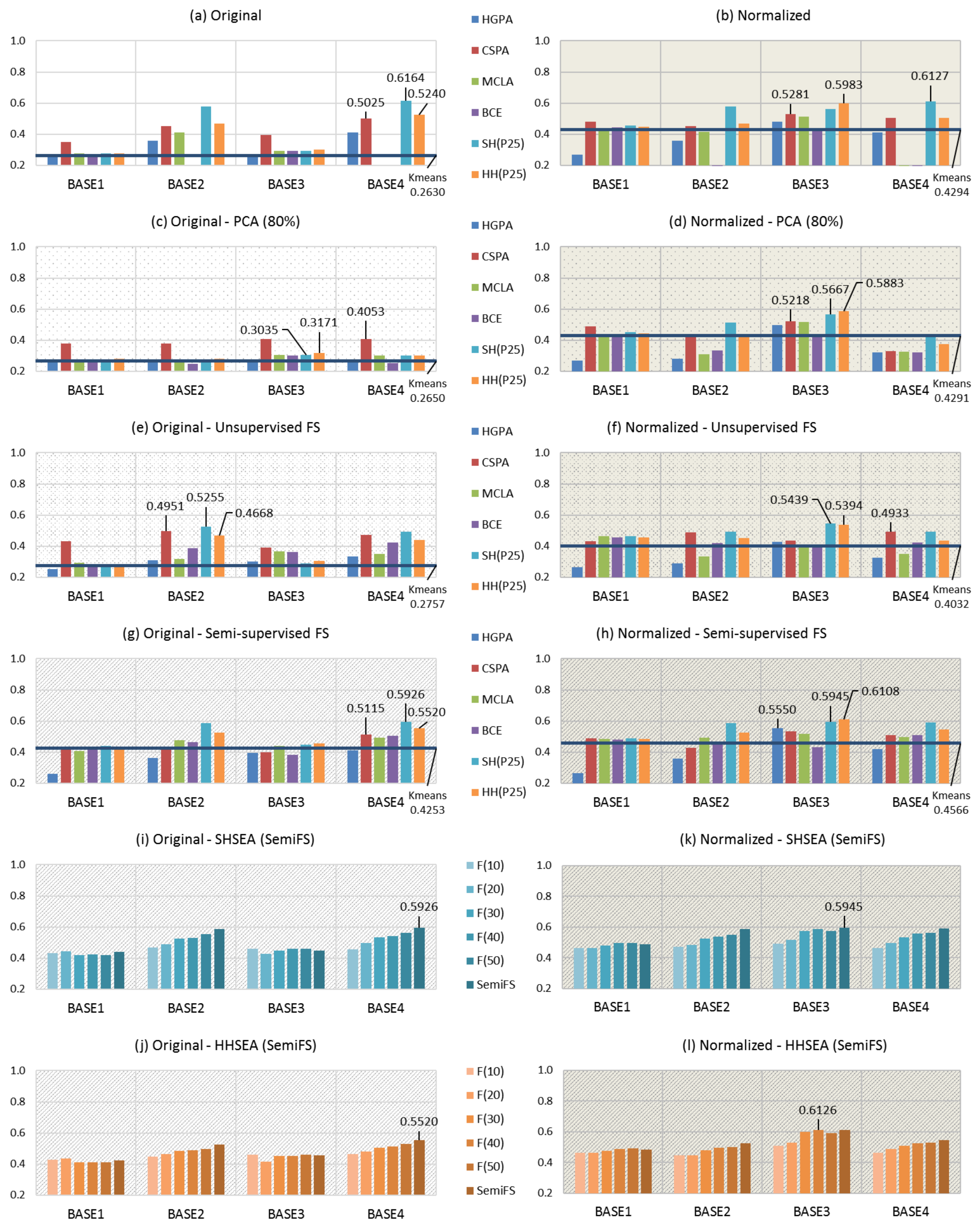


Figure 5.23: Feature extracted vs. Feature selected DWALabSet7

Table 5.10: Comparison of unsupervised mode and semi-supervised mode using extracted features of normalized data sets

Data Sets	Kmeans	Unsupervised				SHSEA			HHSEA		
(Normalized)	MP	Winner	Base Set	MP	Improved %	Base Set	MP	Improved %	Base Set	MP	Improved %
Ionosphere	0.7111	MCLA	4	0.8080	9.7%	3	0.8674	15.6%	**3**	**0.8956**	**18.4%**
Pima	0.6623	MCLA	2	0.6810	1.9%	**2**	**0.7081**	**4.6%**	3	0.6887	2.6%
Balance	0.5498	BCE	1	0.5960	4.6%	3	0.6959	14.6%	**3**	**0.7212**	**17.1%**
Wine	0.9357	HGPA	3	0.9533	1.8%	**3**	**0.9460**	**1.0%**	1	0.9406	0.5%
DWALabSet1	0.6686	BCE	4	0.8080	13.9%	3	0.8674	19.9%	**3**	**0.8956**	**22.7%**
DWALabSet2	0.5612	HGPA	3	0.7247	16.4%	3	0.7367	17.6%	**3**	**0.7740**	**21.3%**
DWALabSet3	0.6126	HGPA	3	0.8127	20.0%	**3**	**0.8320**	**21.9%**	3	0.8287	21.6%
DWALabSet4	0.4991	MCLA	1	0.5350	3.6%	3	0.6804	18.1%	**3**	**0.6929**	**19.4%**
DWALabSet5	0.5596	MCLA	3	0.6587	9.9%	3	0.7156	15.6%	**3**	**0.7302**	**17.1%**
DWALabSet6	0.4521	HGPA	3	0.5742	12.2%	3	0.6222	17.0%	**3**	**0.6502**	**19.8%**
DWALabSet7	0.4259	CSPA	3	0.5410	11.5%	3	0.5640	13.8%	**3**	**0.5847**	**15.9%**

Table 5.11: Micro-precision of K-means using features selected from original data sets by different feature selection methods

Data Sets	No. of all features	MPs	No. of selected features	Feature Selection method	
				Unsupervised	Semi-supervised
Ionosphere	34	0.7111	5	0.6447	0.6941
Pima	8	0.6532	1	0.6514	0.7107
Balance	4	0.5498	1	0.5149	0.5084
Wine	13	0.7007	2	0.5397	0.8242
DWALabSet1	705	0.5033	87	0.5033	0.6129
DWALabSet2	705	0.5033	94	0.5033	0.7484
DWALabSet3	705	0.5367	84	0.5033	0.7683
DWALabSet4	705	0.3400	100	0.3438	0.4272
DWALabSet5	705	0.3600	100	0.3600	0.5272
DWALabSet6	705	0.3474	94	0.3400	0.4445
DWALabSet7	705	0.2630	91	0.2653	0.3999

5.2.4 Selected Features

Feature selection is another common technique used to reduce data dimensionality. To study the effect of feature selection on the performance of clustering algorithms, we pre-process each data set using different feature selection algorithms, including the unsupervised method proposed in [86] and the semi-supervised method proposed in Section 4.2.3. The micro-precision of K-means using selected features of original data sets is listed in Table 5.11. The performance of K-means has not been improved by the unsupervised feature selection algorithm, while the performance of K-means has been improved by the semi-supervised feature selection algorithm on most of the data sets.

Normalized data sets are examined in a similar manner and the micro-precision of K-means is listed in Table 5.12. The performance of K-means has been improved by the unsupervised feature selection on DWALab data sets, while the performance of K-means has been further improved by the semi-supervised feature selection method.

From Table 5.11 and Table 5.12 we notice that about a hundred of features are

Table 5.12: Micro-precision of K-means using features selected from normalized data sets by different feature selection methods

Data Sets	No. of all features	Kmeans	No. of selected features	Feature Selection method	
				Unsupervised	Semi-supervised
Ionosphere	34	0.7111	6	0.6655	0.6904
Pima	8	0.6627	1	0.6345	0.7107
Balance	4	0.5520	1	0.5175	0.5129
Wine	13	0.9389	2	0.6831	0.8603
DWALabSet1	705	0.6628	88	0.6991	0.8372
DWALabSet2	705	0.5609	97	0.6020	0.6554
DWALabSet3	705	0.6120	78	0.8185	0.8387
DWALabSet4	705	0.5058	104	0.4975	0.6658
DWALabSet5	705	0.5452	98	0.6261	0.6236
DWALabSet6	705	0.4513	96	0.4891	0.5157
DWALabSet7	705	0.4294	93	0.4621	0.4763

Table 5.13: Micro-precision of K-means using different numbers of features (from original data sets) selected by the semi-supervised method

Data Sets	All Features	Semi-supervised	No. of Selected Features				
			10	20	30	40	50
DWALabSet1	0.5033	0.8281	0.5406	0.7835	0.7818	0.8112	0.7454
DWALabSet2	0.5033	0.7233	0.5446	0.5881	0.6224	0.6226	0.6199
DWALabSet3	0.5367	0.7353	0.6085	0.7007	0.7173	0.7555	0.7635
DWALabSet4	0.3400	0.5207	0.4124	0.4670	0.5496	0.5598	0.5267
DWALabSet5	0.3600	0.4672	0.4437	0.4939	0.5036	0.6022	0.6138
DWALabSet6	0.3476	0.4177	0.3938	0.4232	0.4683	0.4777	0.4704
DWALabSet7	0.2633	0.4253	0.4243	0.4008	0.4110	0.4435	0.4157

selected for each DWALab data set. To study the effect of the number of selected features on clustering algorithms, we vary the number of features selected by the semi-supervised method. The micro-precision of K-means using different numbers of features selected from the original and normalized data sets is presented in Table 5.13 and Table 5.14 respectively. The optimal number of features to be selected is data dependent and we lack the knowledge on how to determine this number. Instead of determining the optimal number of features to be selected, the micro-precision of K-means using features selected by the proposed semi-supervised feature selection algorithm stays close to the best performance of K-means.

Table 5.14: Micro-precision of K-means using different numbers of features (from normalized data sets) selected by the semi-supervised method

Data Sets (Normalized)	All Features	Semi-supervised	No. of Selected Features 10	20	30	40	50
DWALabSet1	0.6823	0.8364	0.8373	0.8295	0.8337	0.8368	0.8357
DWALabSet2	0.5627	0.6745	0.6906	0.6642	0.6923	0.7063	0.6920
DWALabSet3	0.6207	0.8173	0.8222	0.8153	0.8171	0.8200	0.8230
DWALabSet4	0.5051	0.6334	0.5821	0.6242	0.6350	0.6534	0.6607
DWALabSet5	0.5417	0.6061	0.5522	0.6102	0.6292	0.6264	0.6395
DWALabSet6	0.4556	0.5377	0.5443	0.5367	0.5260	0.5367	0.5317
DWALabSet7	0.4326	0.4566	0.4513	0.4658	0.4662	0.4777	0.4705

To study the effect of feature selection on clustering ensemble methods, we sent different sets of selected features to the proposed system. The performance of the system is illustrated in Figure 5.13 to Figure 5.23. Sub-figures (e) and (f) of each figure represent the system performance using features selected by the unsupervised method. Sub-figures (g) and (h) of each figure represent the system performance using features selected by the semi-supervised method. To study the effect of number of selected features on clustering ensemble methods, we vary the number of selected features and repeat the experiments on DWALab data sets. The performance of SHSEA and HHSEA is illustrated by sub-figures (i) to (l) of Figure 5.17 to Figure 5.23.

The winner of unsupervised methods, its corresponding winning set of base clusterings and the winning set of SHSEA and HHSEA are summarized in Table 5.15 and Table 5.16. The winning set of base clusterings is data dependent. Although the performance of K-means can be improved by using selected features, the system performance using selected features stays close to the system performance using all features. Similar to feature extraction, the advantage of feature selection in semi-supervised clustering ensemble is mainly the reduction of data dimensionality.

Table 5.15: Comparison of unsupervised mode and semi-supervised mode using original features selected by semi-supervised feature selection algorithm

Data Sets	Kmeans	Unsupervised				SHSEA			HHSEA		
(Original)	SemiFS	Winner	Base Set	MP	Improved %	Base Set	MP	Improved %	Base Set	MP	Improved %
Ionosphere	0.6919	HGPA	4	0.7756	8.4%	3	0.8094	11.8%	**3**	**0.8183**	**12.6%**
Pima	0.6515	MCLA	2	0.7107	5.9%	**2**	**0.7180**	**6.6%**	2	0.7180	6.6%
Balance	0.5479	MCLA	1	0.5485	0.1%	2	0.5389	-0.9%	**1**	**0.5514**	**0.3%**
Wine	0.6897	MCLA	1	0.8554	16.6%	**1**	**0.8511**	**16.1%**	1	0.8511	16.1%
DWALabSet1	0.8281	**HGPA**	**4**	**0.8242**	**-0.4%**	1	0.8038	-2.4%	1	0.8142	-1.4%
DWALabSet2	0.7233	MCLA	4	0.7619	3.9%	**4**	**0.7757**	**5.2%**	4	0.7658	4.2%
DWALabSet3	0.7353	CSPA	4	0.7851	5.0%	**4**	**0.7892**	**5.4%**	4	0.7524	1.7%
DWALabSet4	0.5207	CSPA	4	0.6214	10.1%	2	0.6622	14.1%	**4**	**0.6651**	**14.4%**
DWALabSet5	0.4672	CSPA	4	0.6794	21.2%	**4**	**0.6794**	**21.2%**	4	0.6556	18.8%
DWALabSet6	0.4177	CSPA	4	0.5972	17.9%	**4**	**0.6092**	**19.1%**	4	0.5864	16.9%
DWALabSet7	0.4253	CSPA	4	0.5115	8.6%	**4**	**0.5926**	**16.7%**	4	0.5520	12.7%

Table 5.16: Comparison of unsupervised mode and semi-supervised mode using normalized features selected by semi-supervised feature selection algorithm

Data Sets	Kmeans	Unsupervised				SHSEA			HHSEA		
(Normalized)	SemiFS	Winner	Base Set	MP	Improved %	Base Set	MP	Improved %	Base Set	MP	Improved %
Ionosphere	0.6993	HGPA	4	0.7689	7.0%	3	0.8150	11.6%	**3**	**0.8309**	**13.2%**
Pima	0.6693	MCLA	2	0.7107	4.1%	**2**	**0.7184**	**4.9%**	2	0.7184	4.9%
Balance	0.5525	MCLA	1	0.5403	-1.2%	2	0.5557	0.3%	**1**	**0.5672**	**1.5%**
Wine	0.8653	MCLA	1	0.8116	-5.4%	1	0.8367	-2.9%	**1**	**0.8368**	**-2.8%**
DWALabSet1	0.8364	HGPA	3	0.8540	1.8%	3	0.8557	1.9%	**3**	**0.8583**	**2.2%**
DWALabSet2	0.6745	MCLA	4	0.7648	9.0%	**4**	**0.7748**	**10.0%**	4	0.7697	9.5%
DWALabSet3	0.8173	**CSPA**	**4**	**0.8748**	**5.7%**	4	0.8679	5.1%	3	0.8638	4.6%
DWALabSet4	0.6334	MCLA	1	0.6760	4.3%	**3**	**0.6788**	**4.5%**	3	0.6738	4.0%
DWALabSet5	0.6061	CSPA	4	0.6759	7.0%	3	0.6937	8.8%	**3**	**0.7053**	**9.9%**
DWALabSet6	0.5377	CSPA	4	0.6096	7.2%	3	0.6453	10.8%	**4**	**0.6509**	**11.3%**
DWALabSet7	0.4566	HGPA	3	0.5550	9.8%	3	0.5945	13.8%	**3**	**0.6108**	**15.4%**

Table 5.17: Data Information II: the number of training and testing data points, features, classes and subsets

Data Sets	Training points	Testing point	Features	Classes	Subsets(Q)
Segmentation	210	2100	19	7	2
DWALabSet8	134	2684	705	2	4

5.2.5 Distributed Data Observations

In this section, we evaluate the performance of parallel distributed clustering system (proposed in Section 4.6) using one data set from UCI [96] and one data set from DWALab. The information of these data sets is listed in Table 5.17.

We first conduct the series of experiments described in Section 5.2.1 and Section 5.2.2 using the entire data sets, original and normalized. We then divide "Segmentation" data set into two subsets (i.e., $Q = 2$) and divide "DWALabSet8" into four subsets (i.e., $Q = 4$) to evaluate the proposed parallel system. The micro-precision of K-means is listed in Table 5.18. Note that "$Q = 1$" represents the experiments running on the entire data sets. The performance of the proposed distributed clustering system (unsupervised mode and semi-supervised mode) and the performance of the parallel system are illustrated by Figure 5.24 and Figure 5.25. Note that "SH(Q1)" represents the performance of SHSEA when clustering the entire data set, while "SH(Q2)" represents the performance of SHSEA when the data set is divided into two subsets, i.e., $Q = 2$.

Similar to the system performance on data sets listed in Table 5.2, "BASE3" is the winning set of base clusterings when the entire normalized data sets are the input of the proposed system. The semi-supervised algorithms perform better than unsupervised algorithms on these two data sets. The performance of the parallel system

Table 5.18: Micro-precision of K-means using the entire data sets ($Q = 1$) and each subset separately

Segmentation	Q=1	Q=2			
		q1		q2	
Original	0.4994	0.4951		0.4934	
Normalized	0.5945	0.5937		0.6057	
DWALabSet8	Q=1	Q=4			
		q1	q2	q3	q4
Original	0.5014	0.5128	0.5061	0.5220	0.5148
Normalized	0.5494	0.5749	0.5690	0.5521	0.5686

is very closed to the performance of the semi-supervised mode of the distributed clustering system. Therefore, from the performance aspect the parallel system is as good as the distributed clustering system.

Recall that clustering ensemble methods usually consist of the base clustering generation step and the consensus fusion steps. Therefore, the computational time of an ensemble method is determined by the time of generating and combining base clusterings. In this thesis all experiments were performed using the software package MATLAB R2016b on an Acer PC with processor AMD A8-3800 2.40 GHz and RAM 8GB. In the proposed parallel system, the subsets of input data are processed in parallel. Therefore, its computational time is determined by the longest time of generating and combining base clusterings among all subsets. The computational time is listed in Table 5.19 to Table 5.22, measured in seconds. These tables show that from the computational time aspect the proposed parallel distributed clustering system outperforms on both data sets.

5.2.6 Detection of a New Cluster

Recall that DWAlab data set is obtained from the study of human breast cancer cells undergoing treatment of different drugs. When a certain type of drug is injected into

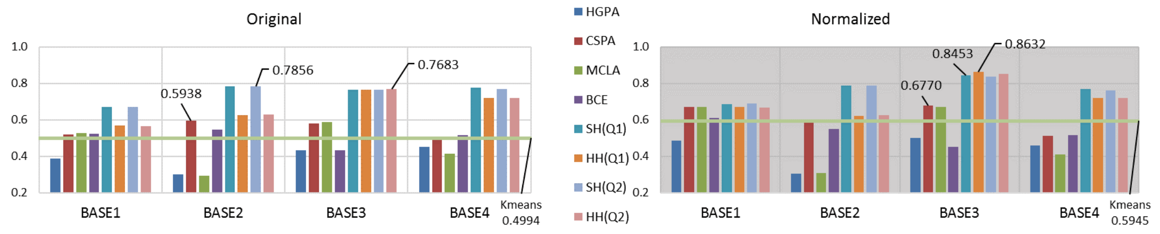


Figure 5.24: Original vs. Normalized Segmentation

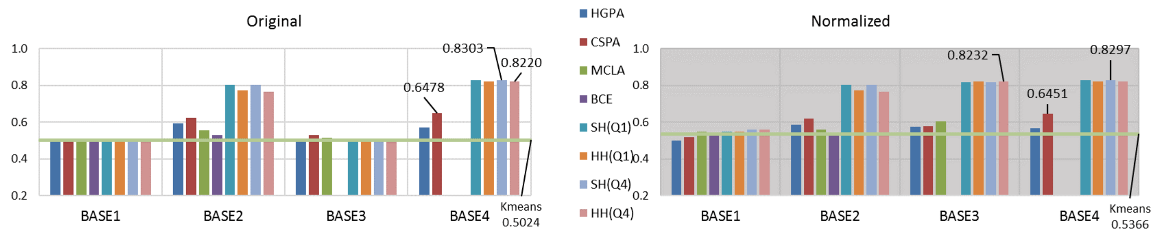


Figure 5.25: Original vs. Normalized DWALabSet8

Table 5.19: Comparison of computational time of distributed clustering system and parallel distributed clustering system: original Segmentation

Original	BASE1			BASE2			BASE3			BASE4		
	BCG	Ensemble	Total	BCG	Ensemble	Total	BCG	Ensemble	Total	BCG	Ensemble	Total
HGPA	0.551	0.547	1.098	0.294	0.553	0.847	1.245	1.330	2.574	0.514	1.411	1.925
CSPA		4.349	4.900		9.530	9.825		3.365	4.610		9.164	9.678
MCLA		0.536	1.087		0.449	0.743		0.880	2.124		0.866	1.380
BCE		38.006	38.557		37.457	37.751		127.047	128.292		178.462	178.976
SHSEA	0.583	1.317	1.900	0.303	1.244	1.548	1.335	1.266	2.601	0.533	1.152	1.686
HHSEA		1.826	2.409		1.610	1.914		1.763	3.098		1.568	2.102
SH(Q2)	0.452	0.793	1.244 (34.5%)	0.210	0.660	0.870 (43.8%)	0.788	0.718	1.506 (42.1%)	0.366	0.620	0.986 (41.5%)
q1	0.451	0.783		0.210	0.658		0.788	0.718		0.366	0.615	
q2	0.452	0.793		0.207	0.660		0.772	0.700		0.360	0.620	
HH(Q2)	0.452	1.074	1.526 (36.7%)	0.210	0.860	1.070 (44.1%)	0.788	0.974	1.762 (43.1%)	0.366	0.840	1.207 (42.6%)
q1	0.451	1.040		0.210	0.851		0.788	0.974		0.366	0.840	
q2	0.452	1.074		0.207	0.860		0.772	0.974		0.360	0.835	

Table 5.20: Comparison of computational time of distributed clustering system and parallel distributed clustering system: normalized Segmentation

Normalized	BASE1			BASE2			BASE3			BASE4		
	BCG	Ensemble	Total	BCG	Ensemble	Total	BCG	Ensemble	Total	BCG	Ensemble	Total
HGPA	0.573	0.593	1.166	0.261	0.553	0.814	0.863	1.317	2.180	0.488	1.207	1.695
CSPA		3.861	4.434		9.530	9.791		2.316	3.179		8.927	9.416
MCLA		0.538	1.111		0.449	0.709		0.605	1.468		0.754	1.243
BCE		27.041	27.614		37.457	37.718		112.388	113.251		165.477	165.965
SHSEA	0.632	1.347	1.979	0.275	1.244	1.520	0.900	1.207	2.107	0.502	1.144	1.646
HHSEA		1.812	2.444		1.610	1.886		1.671	2.571		1.567	2.069
SH(Q2)	0.382	0.818	1.200 (39.4%)	0.189	0.657	0.847 (44.3%)	0.565	0.662	1.227 (41.8%)	0.375	0.649	1.024 (37.8%)
q1	0.382	0.818		0.189	0.653		0.565	0.662		0.375	0.648	
q2	0.358	0.811		0.185	0.657		0.549	0.661		0.370	0.649	
HH(Q2)	0.382	1.088	1.470 (39.9%)	0.189	0.851	1.040 (44.8%)	0.565	0.912	1.477 (42.5%)	0.375	0.896	1.271 (38.6%)
q1	0.382	1.075		0.189	0.847		0.565	0.912		0.375	0.896	
q2	0.358	1.088		0.185	0.851		0.549	0.912		0.370	0.881	

Table 5.21: Comparison of computational time of distributed clustering system and parallel distributed clustering system: original DWALabSet8

Original	BASE1			BASE2			BASE3			BASE4		
	Base	Ensemble	Total	Base	Ensemble	Total	Base	Ensemble	Total	Base	Ensemble	Total
HGPA	2.327	0.568	1.136	6.179	17.204	23.383	9.744	0.898	10.642	30.508	143.397	173.905
CSPA		16.254	16.822		24.623	30.801		16.742	26.486		151.236	181.744
MCLA		0.418	0.986		15.588	21.767		0.944	10.688		n/a	n/a
BCE		6.788	7.356		540.665	546.843		n/a	n/a		n/a	n/a
SHSEA	2.422	0.589	3.011	6.284	16.952	23.237	10.289	0.579	10.868	31.985	18.345	50.329
HHSEA		1.478	3.900		32.632	38.917		1.062	11.351		34.710	66.695
SH(Q4)	0.869	0.179	1.048 (65.2%)	3.891	4.654	8.545 (63.2%)	3.745	0.169	3.914 (64.0%)	11.733	4.928	16.661 (66.9%)
q1	0.795	0.175		3.876	4.651		3.646	0.168		11.733	4.925	
q2	0.819	0.178		3.891	4.654		3.661	0.166		11.723	4.928	
q3	0.819	0.172		3.870	4.652		3.745	0.167		11.714	4.924	
q4	0.869	0.179		3.871	4.651		3.593	0.169		11.724	4.914	
HH(Q4)	0.869	0.415	1.284 (67.1%)	3.891	9.046	12.937 (66.8%)	3.745	0.308	4.053 (64.3%)	11.733	9.335	21.068 (68.4%)
q1	0.795	0.398		3.876	9.010		3.646	0.301		11.733	9.334	
q2	0.819	0.415		3.891	9.010		3.661	0.306		11.723	9.335	
q3	0.819	0.377		3.870	9.014		3.745	0.308		11.714	9.323	
q4	0.869	0.409		3.871	9.046		3.593	0.305		11.724	9.334	

Table 5.22: Comparison of computational time of distributed clustering system and parallel distributed clustering system: normalized DWALabSet8

Normalized	BASE1			BASE2			BASE3			BASE4		
	Base	Ensemble	Total	Base	Ensemble	Total	Base	Ensemble	Total	Base	Ensemble	Total
HGPA	12.342	0.607	12.949	6.053	17.360	23.412	29.818	0.919	30.736	26.710	138.809	165.519
CSPA		16.141	28.483		24.825	30.878		9.272	39.090		147.759	174.469
MCLA		0.517	12.859		15.664	21.717		1.031	30.849		n/a	n/a
BCE		10.961	23.303		476.930	482.983		n/a	n/a		n/a	n/a
SHSEA	12.193	0.586	12.779	6.126	16.317	22.443	32.070	0.565	32.635	28.125	18.088	46.212
HHSEA		1.096	13.289		33.575	39.700		1.031	33.101		33.062	61.187
SH(Q4)	3.712	0.171	3.883 (69.6%)	3.841	4.474	8.315 (62.9%)	4.523	0.159	4.682 (85.7%)	11.305	6.193	17.498 (62.1%)
q1	3.347	0.169		3.832	4.460		4.430	0.157		11.284	6.179	
q2	3.712	0.170		3.841	4.471		4.456	0.155		11.305	6.180	
q3	3.426	0.171		3.830	4.465		4.523	0.159		11.278	6.179	
q4	3.066	0.169		3.828	4.474		4.412	0.156		11.290	6.193	
HH(Q4)	3.712	0.322	4.034 (69.6%)	3.841	9.191	13.032 (67.2%)	4.523	0.289	4.812 (85.5%)	11.305	10.496	21.802 (64.4%)
q1	3.347	0.318		3.832	9.152		4.430	0.286		11.284	10.482	
q2	3.712	0.318		3.841	9.171		4.456	0.283		11.305	10.490	
q3	3.426	0.315		3.830	9.162		4.523	0.289		11.278	10.486	
q4	3.066	0.322		3.828	9.191		4.412	0.285		11.290	10.496	

cancer cells, the cells usually react differently: a portion of the cells may slightly react to the injected drug (such as slightly enlarged); another portion of the cells may react strongly (such as loss of nucleus); and the rest may not react to the injected drug at all. For those cells that strongly react to the injected drug, it is very likely that their statistical properties vary significantly and they can form a new cluster. Therefore, in the study of the effect of a certain drug to cancer cells, we could apply our proposed new cluster detection algorithm to automatically detect the existence of cancer cells that strongly react to the injected drug.

In this section, we provide numerical examples to show the performance of the proposed new cluster detection algorithm. The original test files contain data observations from different classes. Each original test file has a fixed amount of training data. To evaluate the new cluster detection algorithm, we insert additional data points to the original test files and vary the number of additional data points. To evaluate the probability of successful detection of a new cluster, we insert a mixture of data points from a new class and from existing classes and vary the proportion of the data points from a new class. For each original test file and a particular number of additional points, we randomly generate 20 versions of additional data set $\mathbf{X}_a$ using one of the mixture proportions listed in Table 5.23. The number of total successful detections of a new cluster is provided in Table 5.24. As expected, the probability of successful detections of a new cluster using the proposed algorithm goes higher when the number of data points from a new class increases.

One of the challenging problems for future research would obviously include study in which the probability of successful detection would be analyzed as a function of the mixing proportion. It could provide useful information on designing the experiments

Table 5.23: Mixing proportion of data points from a new class and existing classes

Mixing Proportion	Proportion of data points from	
	A new class	Existing classes
Type 1	1	0
Type 2	2/3	1/3
Type 3	1/2	1/2
Type 4	1/3	2/3
Type 5	0	1

to examine the effect of certain drugs to the cancer cells.

Table 5.24: Number of successful detection of a new cluster when different amount of additional data points are added to the original data sets

P = 15%	No. of data points		No. of detection of a new cluster					Total Success
	Original	Added	Type 1 (/20)	Type 2 (/20)	Type 3 (/20)	Type 4 (/20)	Type 5 (/20)	(/100)
2ClassesAdd1	257	50	20	17	17	12	3	83
		100	20	20	17	15	1	91
		150	20	20	19	15	2	92
3ClassesAdd1	518	50	20	15	12	9	2	74
		100	20	18	18	8	1	83
		150	20	20	19	9	3	85

Chapter 6

Conclusions

In this thesis, we have proposed information fusion methods for the detection, classification and clustering applications.

In distributed detection, the detection error can be further reduced by considering the correlation between local decisions. We have proposed a sub-optimal method to estimate the unknown probabilities of the hypotheses, anomalies of local detectors and correlation coefficients. These unknown parameters are necessary for the fusion center to make final decisions. We provided numerical examples to demonstrate the improvement of the system performance when the correlation between local decisions is considered.

By viewing the M categories pre-defined for a classification task as M hypotheses in the detection problem, we can simply apply the information fusion techniques used in detection problem to combine multiple classification assignments in order to improve the classification results.

In distributed clustering, we have proposed semi-supervised clustering ensemble algorithms based on utilizing labelled training data to improve the clustering results.

Analogous to distributed detection system, we have proposed a distributed clustering system, which consists of a base clustering generator and a fusion center. In this thesis, we designed four different base clustering generators and two operational modes for the fusion center. We provided numerical examples to demonstrate the performance of the proposed system and the effect of base clusterings on the clustering ensemble methods. When data to be clustered contains features measured in diverse ranges, it is possible to improve the performance of the proposed distributed clustering system by generating base clusterings using features individually. We also evaluated the performance of the proposed system using normalized data sets, extracted features and selected features of each data sets. The empirical study showed that for data sets containing features measured in diverse ranges the system performance can be improved by normalization. Feature extraction using PCA is able to reduce the data dimensionality, but it is not able to improve the performance of clustering and clustering ensemble methods. Although feature selection is able to improve the performance of clustering algorithms, it is not able to further improve the performance of clustering ensemble methods.

To efficiently perform cluster analysis on high-volume data sets, we proposed a parallel distributed clustering system that consists of several distributed clustering system connected in parallel. The major objective of this parallel system is to reduce the computational time for clustering large size data set. We provided numerical examples to demonstrate the performance of the proposed parallel system and the computational times to clustering large data sets. To provide feedback to the system when new data observations are available, we proposed a new cluster detection algorithm to detect the occurrence of new data points (other than existing training classes). The role of this feedback system is to request additional training labels or switch from the semi-supervised mode to the unsupervised mode when a new cluster is detected.

In the near future we will focus on the following research topics:

1. To further study the correlation between local decisions, we will use multi-channel EEG signals instead of the single channel EEG signals mentioned in this thesis.

2. We will study the effect of correlation in clustering ensemble problem.

3. We will study the effect of selection of reference points.

4. To improve the accuracy of HHSEA, we will use other fusion algorithms rather than majority voting.

5. For the purpose of studying the effect of certain drugs to the cancer cells, we will conduct more experiments using the new cluster detection algorithm to study the successful probability of detection of a new class as a function of the mixing proportion of data observations from a new class and existing classes.

Bibliography

[1] H. Li and A. Jeremić, "Neonatal seizure detection using blind multichannel information fusion," in *Proc. IEEE ICASSP*, Prague, Czech Republic, 2011, pp. 649–652.

[2] D. Hall and J. Llinas, *Multisensor Data Fusion.* CRC press, 2001.

[3] R. S. Blum and Z. Liu, *Multi-sensor Image Fusion and its Applications.* CRC press, 2005.

[4] V. Torra and Y. Narukawa, *Modeling Decisions: Information Fusion and Aggregation Operators.* Springer Science & Business Media, 2007.

[5] D. Mandic, M. Golz, A. Kuh, D. Obradovic, and T. Tanaka, *Signal Processing Techniques for Knowledge Extraction and Information Fusion.* Springer, 2008.

[6] E. Cambria, D. Rajagopal, D. Olsher, and D. Das, "Big social data analysis," *Big Data Computing*, pp. 401–414, 2013.

[7] K. F. Wallis, "Combining forecasts–forty years later," *Applied Financial Economics*, vol. 21, no. 1-2, pp. 33–41, 2011.

[8] A. Ross and A. Jain, “Information fusion in biometrics,” *Pattern Recogn. Lett.*, vol. 24, no. 13, pp. 2115–2125, 2003.

[9] P. K. Varshney, *Distributed Detection and Data Fusion.* Springer-Verlag New York, Inc., 1996.

[10] R. Viswanathan and P. K. Varshney, “Distributed detection with multiple sensors: part I - fundamentals,” *Proc. IEEE*, vol. 85, no. 1, pp. 54–63, 1997.

[11] Z. Chair and P. K. Varshney, “Optimal data fusion in multiple sensor detection systems,” *IEEE Trans. Aerosp. Electron. Syst.*, no. 1, pp. 98–101, 1986.

[12] A. Naim and M. Kam, “On-line estimation of probabilities for distributed Bayesian detection,” *Automatica*, vol. 30, no. 4, pp. 633–642, 1994.

[13] G. Mirjalily, Z.-Q. Luo, T. N. Davidson, and E. Bosse, “Blind adaptive decision fusion for distributed detection,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 1, pp. 34–52, 2003.

[14] N. Ansari, E. S. Hou, B.-O. Zhu, and J.-G. Chen, “Adaptive fusion by reinforcement learning for distributed detection systems,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 32, no. 2, pp. 524–531, 1996.

[15] N. Ansari, J.-G. Chen, and Y.-Z. Zhang, “Adaptive decision fusion for unequiprobable sources,” *IEEE Proceedings Radar, Sonar and Navigation*, vol. 144, no. 3, pp. 105–111, 1997.

[16] W. Baek, “Optimal M-ary data fusion with distributed sensors,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 31, no. 3, pp. 1150–1152, 1995.

[17] B. Liu, A. Jeremić, and K. M. Wong, "Optimal distributed detection of multiple hypotheses using blind algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 47, no. 1, pp. 317–331, 2011.

[18] Q. Zhang and P. K. Varshney, "Decentralized M-ary detection via hierarchical binary decision fusion," *Inform. Fusion*, vol. 2, no. 1, pp. 3–16, 2001.

[19] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, 1998.

[20] R. Ranawana and V. Palade, "Multi-classifier systems: Review and a roadmap for developers," *Int. J. Hybrid Intell. Syst.*, vol. 3, no. 1, pp. 35–61, 2006.

[21] R. J. Urbanowicz and J. H. Moore, "Learning classifier systems: a complete introduction, review, and roadmap," *Artificial Evolution and Applications*, vol. 2009, p. 1, 2009.

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

[23] R. Xu and D. Wunsch, *Clustering.* John Wiley & Sons, 2008, vol. 10.

[24] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *Mach. Learn. Res.*, vol. 3, pp. 583–617, 2003.

[25] X. Z. Fern and C. E. Brodley, "Random projection for high dimensional data clustering: A cluster ensemble approach," in *Proc. Int. Conf. Machine Learning*, vol. 3, Washington, DC, USA, 2003, pp. 186–193.

[26] H. Wang, H. Shan, and A. Banerjee, "Bayesian cluster ensembles," *Statistical Analysis and Data Mining*, vol. 4, no. 1, pp. 54–70, 2011.

[27] R. Ghaemi, M. N. Sulaiman, H. Ibrahim, and N. Mustapha, "A survey: clustering ensembles techniques," *World Academy of Science, Engineering and Technology*, vol. 50, pp. 636–645, 2009.

[28] S. Vega-Pons and J. Ruiz-Shulcloper, "A survey of clustering ensemble algorithms," *Int. J. Pattern Recog. Artificial Intell.*, vol. 25, no. 03, pp. 337–372, 2011.

[29] S. Dudoit and J. Fridlyand, "Bagging to improve the accuracy of a clustering procedure," *Bioinformatics*, vol. 19, no. 9, pp. 1090–1099, 2003.

[30] A. L. Fred and A. K. Jain, "Combining multiple clusterings using evidence accumulation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 835–850, 2005.

[31] E. Dimitriadou, A. Weingessel, and K. Hornik, "Voting-merging: An ensemble method for clustering," in *Int. Conf. Artificial Neural Networks.* Springer, 2001, pp. 217–224.

[32] A. Topchy, A. K. Jain, and W. Punch, "A mixture model for clustering ensembles," in *Proc. SIAM Int. Conf. Data Mining.* SIAM, 2004, pp. 379–390.

[33] M. Kam, Q. Zhu, and W. S. Gray, "Optimal data fusion of correlated local decisions in multiple sensor detection systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 28, no. 3, pp. 916–920, 1992.

[34] H. Li, “Neonatal seizure detection using blind adaptive fusion,” Master’s thesis, Dept. Elec. Eng., McMaster Univ., Ontario, Canada, 2010.

[35] J. Volpe, *Neurology of the Newborn.* WB Saunders Co, 2001.

[36] S. Faul, G. Boylan, S. Connolly, L. Marnane, and G. Lightbody, “An evaluation of automated neonatal seizure detection methods,” *Clinical Neurophysiology*, vol. 116, no. 7, pp. 1533–1541, 2005.

[37] A. Liu, J. Hahn, G. Heldt, and R. Coen, “Detection of neonatal seizures through computerized EEG analysis,” *Electroencephalography and Clinical Neurophysiology*, vol. 82, no. 1, pp. 30–37, 1992.

[38] J. Gotman, D. Flanagan, J. Zhang, and B. Rosenblatt, “Automatic seizure detection in the newborn: methods and initial evaluation,” *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 3, pp. 356–362, 1997.

[39] P. Celka and P. Colditz, “A computer-aided detection of EEG seizures in infants: asingular-spectrum approach and performance comparison,” *IEEE Trans. Biomed. Eng.*, vol. 49, no. 5, pp. 455–462, 2002.

[40] H. L. Van Trees, *Detection Estimation and Modulation Theory, pt. I.* Wiley, 1968.

[41] D. Jiang, C. Tang, and A. Zhang, “Cluster analysis for gene expression data: a survey,” *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 11, pp. 1370–1386, 2004.

[42] P. Berkhin, “A survey of clustering data mining techniques,” in *Grouping Multidimensional Data.* Springer, 2006, pp. 25–71.

[43] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recogn. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[44] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 3, pp. 267–279, 2014.

[45] C. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on big data," *Information Sciences*, vol. 275, pp. 314–347, 2014.

[46] A. McAfee, E. Brynjolfsson *et al.*, "Big data: the management revolution," *Harvard Business Review*, vol. 90, no. 10, pp. 60–68, 2012.

[47] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 97–107, 2014.

[48] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of big data on cloud computing: Review and open research issues," *Information Systems*, vol. 47, pp. 98–115, 2015.

[49] C.-W. Tsai, C.-F. Lai, H.-C. Chao, and A. V. Vasilakos, "Big data analytics: a survey," *Big Data*, vol. 2, no. 1, p. 21, 2015.

[50] G. Bello-Orgaz, J. J. Jung, and D. Camacho, "Social big data: Recent achievements and new challenges," *Information Fusion*, vol. 28, pp. 45–59, 2016.

[51] E. Rendón, I. Abundez, A. Arizmendi, and E. Quiroz, "Internal versus external

cluster validation indexes," *Int. J. Computers and Communications*, vol. 5, no. 1, pp. 27–34, 2011.

[52] D. S. Modha and W. S. Spangler, "Feature weighting in k-means clustering," *Mach. Learn.*, vol. 52, no. 3, pp. 217–237, 2003.

[53] B. S. Everitt, D. Stahl, M. Leese, and S. Landau, *Cluster Analysis.* John Wiley & Sons, 2011.

[54] G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications.* SIAM, 2007, vol. 20.

[55] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications.* CRC Press, 2013.

[56] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer*, vol. 32, no. 8, pp. 68–75, 1999.

[57] R. Sharan and R. Shamir, "Click: a clustering algorithm with applications to gene expression analysis," in *Proc. Int. Conf. Intell Syst Mol Biol*, vol. 8, no. 307, 2000, p. 16.

[58] G. H. Ball and D. J. Hall, "Isodata, a novel method of data analysis and pattern classification," DTIC Document, Tech. Rep., 1965.

[59] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern Recogn.*, vol. 36, no. 2, pp. 451–461, 2003.

[60] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley & Sons, 2009, vol. 344.

[61] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984.

[62] C. Fraley and A. E. Raftery, "MCLUST: Software for model-based cluster analysis," *Classification*, vol. 16, no. 2, pp. 297–306, 1999.

[63] V. Estivill-Castro, "Why so many clustering algorithms: a position paper," *ACM SIGKDD Explorations Newsletter*, vol. 4, no. 1, pp. 65–75, 2002.

[64] A. Topchy, A. K. Jain, and W. Punch, "Clustering ensembles: Models of consensus and weak partitions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 12, pp. 1866–1881, 2005.

[65] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[66] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM J. Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998.

[67] V. I. Voloshin, *Introduction to Graph and Hypergraph Theory.* Nova Science Publ., 2009.

[68] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised Learning.* MIT press Cambridge, 2006.

[69] A. Demiriz, K. P. Bennett, and M. J. Embrechts, "Semi-supervised clustering using genetic algorithms," *Artificial Neural Networks in Engineering*, pp. 809–814, 1999.

[70] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *Proc. 19th Int. Conf. Machine Learning*, 2002.

[71] J. Sinkkonen and S. Kaski, "Clustering based on conditional distributions in an auxiliary space," *Neural Computation*, vol. 14, no. 1, pp. 217–239, 2002.

[72] D. Cohn, R. Caruana, and A. McCallum, "Semi-supervised clustering with user feedback," *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, vol. 4, no. 1, pp. 17–32, 2003.

[73] Y. Liu, R. Jin, and A. K. Jain, "Boostcluster: Boosting clustering by pairwise constraints," in *Proc. 13th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. ACM, 2007, pp. 450–459.

[74] F. Wang, T. Li, and C. Zhang, "Semi-supervised clustering via matrix factorization," in *Proc. SIAM Int. Conf. Data Mining*. SIAM, 2008, pp. 1–12.

[75] A. M. Iqbal, A. Moh'd, and Z. Khan, "Semi-supervised clustering ensemble by voting," *arXiv preprint arXiv:1208.4138*, 2012.

[76] H. Liu and H. Motoda, *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Springer Science & Business Media, 1998.

[77] D. Pyle, *Data Preparation for Data Mining*. Morgan Kaufmann, 1999, vol. 1.

[78] M. C. de Souto, D. S. de Araujo, I. G. Costa, R. G. Soares, T. B. Ludermir, and A. Schliep, "Comparative study on normalization procedures for cluster analysis of gene expression datasets," in *IEEE Int. Joint Conf. Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 2792–2798.

[79] N. K. Visalakshi and K. Thangavel, "Impact of normalization in distributed k-means clustering," *Int. J. Soft Computing*, vol. 4, no. 4, pp. 168–172, 2009.

[80] R. J. Larsen, M. L. Marx *et al.*, *An Introduction to Mathematical Statistics and its Applications*. Prentice-Hall Englewood Cliffs, NJ, 1986, vol. 2.

[81] G. W. Milligan and M. C. Cooper, "A study of standardization of variables in cluster analysis," *Classification*, vol. 5, no. 2, pp. 181–204, 1988.

[82] I. Jolliffe, *Principal Component Analysis*. Wiley Online Library, 2002.

[83] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[84] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[85] V. De Silva and J. B. Tenenbaum, "Global versus local methods in nonlinear dimensionality reduction," *Advances in Neural Information Processing Systems*, pp. 721–728, 2003.

[86] P. Mitra, C. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 301–312, 2002.

[87] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proc. Int. Conf. Machine Learning*, vol. 3, 2003, pp. 856–863.

[88] R. Varshavsky, A. Gottlieb, M. Linial, and D. Horn, “Novel unsupervised feature filtering of biological data,” *Bioinformatics*, vol. 22, no. 14, pp. e507–e513, 2006.

[89] R. Kohavi and G. H. John, “Wrappers for feature subset selection,” *Artificial Intell.*, vol. 97, no. 1, pp. 273–324, 1997.

[90] S. K. Pal, R. K. De, and J. Basak, “Unsupervised feature evaluation: a neuro-fuzzy approach,” *IEEE Trans. Neural Netw.*, vol. 11, no. 2, pp. 366–376, 2000.

[91] J. G. Dy and C. E. Brodley, “Feature selection for unsupervised learning,” *Mach. Learn. Res*, vol. 5, no. Aug, pp. 845–889, 2004.

[92] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Mach. Learn. Res.*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[93] H. Liu and H. Motoda, *Computational Methods of Feature Selection.* CRC Press, 2007.

[94] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, “An efficient k-means clustering algorithm: Analysis and implementation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, 2002.

[95] S. Bandyopadhyay and E. J. Coyle, “An energy efficient hierarchical clustering algorithm for wireless sensor networks,” in *22nd Annu. Joint Conf. IEEE Computer and Communications*, vol. 3. IEEE Societies, 2003, pp. 1713–1723.

[96] M. Lichman, “UCI machine learning repository,” 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[97] Z.-H. Zhou and W. Tang, “Clusterer ensemble,” *Knowledge-Based Systems*, vol. 19, no. 1, pp. 77–83, 2006.

[98] R. J. Mehdi, “Frechet means of riemannian distances: Evaluations and applications,” Master’s thesis, Dept. Elec. Eng., McMaster Univ., Ontario, Canada, 2014.