ENHANCEMENT OF SPATIAL DATA ANALYSIS

HU TIANMING

(BSc, NANJING UNIVERSITY, CHINA; MEng, NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2005

Acknowledgment

I am indebted to my supervisor, Dr. Sung Sam Yuan, for his guidance during my doctoral studies.

Thanks also go to the National University of Singapore for providing me with the Research Scholarship.

Last but not least, I would like to thank my family for their support.

Contents

Summary

This thesis studies several problems related to clustering on spatial data. It roughly divides into two parts based on data types. Chapters 2 and 3 concentrate on mixture models for regressing and clustering spatial geographic data, for which the attributes under consideration are explicitly divided into non-spatial normal attributes and spatial attributes that describe the object's location. The second part continues to examine clustering from another two perspectives on general spatial data, for which the distinction between spatial and non-spatial attributes is dropped. At a higher level we explore consensus clustering in Chapter 4. At a finer level we study outlier detection in Chapter 5. These topics are discussed in some detail below.

In Chapter 2, we investigate data fusion in radial basis function (RBF) networks for spatial regression. Regression is linked to clustering via classification. That is, clustering can be regarded as an unsupervised type of classification, which, in turn, is a specialized form of regression with the discrete target variable. Ignoring spatial information, conventional RBF networks usually fail to give satisfactory results on spatial data. In contrast to input fusion, we incorporate spatial information further into RBF networks by fusing output from hidden and output layers. Empirical studies demonstrate the advantage of hidden fusion over others in terms of regression quality. Furthermore, compared to conventional RBF networks, hidden fusion does not entail much extra computation.

In Chapter 3, we propose a Hybrid Expectation-Maximization (HEM) approach for spatial clustering using Gaussian mixture. The goal is to efficiently incorporate spatial information while avoiding much additional computation incurred by Neighborhood Expectation-Maximization (NEM) for E-step. In HEM, early training is performed via a selective hard EM till the penalized likelihood criterion no longer increases. Then

training is turned to NEM, which runs only one iteration of E-step. Thus spatial information is incorporated throughout HEM, which achieves better clustering results than EM and comparable results to NEM. Its complexity is retained between EM and NEM.

In Chapter 4, we continue to study clustering at a higher level. Consensus clustering aims to combine a given set of multiple candidate partitions into a single consolidated partition that is compatible to them. We first propose a series of entropy-based functions for measuring distance among partitions. Then we develop two combining methods for the global optimal partition based on the new similarity between objects determined by the whole candidate set. Given a set of candidate clusterings, under certain conditions, the local/global centroid clustering will be top/middle-ranked in terms of closeness to the true clustering.

In Chapter 5, we turn our attention away from the majority of the data inside clusters to those rare outliers who cannot be assigned to any cluster. Most algorithms target outliers with exceptionally low density, compared to nearby clusters of high density. Besides the pattern of high density clustering, however, we show that there is another pattern, low density regularity. Thus, there are at least two types of corresponding outliers w.r.t. them. We propose two techniques, one used to identify the two patterns and the other used to simultaneously detect outliers w.r.t. them.

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Data Analysis

The terms data analysis and data mining are sometimes used interchangeably. They can be defined as the non-trivial extraction of implicit, previously unknown and potentially useful information and knowledge from data. Data mining is a relatively new jargon used by database researchers, who emphasize the sheer volume of data and provide algorithms that are scalable in terms of both data size and dimensionality.

The entire data analysis/mining process may be illustrated with the following example, where the domain expert, say, a social scientist, consults the data analyst to solve a problem. The social scientist is interested in the explanation of the unusually low voting rate for presidential election in some cities. The ball is now in the court of the data analyst who must decide which techniques to use to address the problem. For instance, he may decide that the problem is best addressed in the framework of regression where voting rate is modeled as a function of relevant demographic variables. He then must choose an appropriate algorithm for implementation, which typically outputs a set of hypotheses (estimated parameters in the regression model). Thus the output is a pattern, which undergoes verification and visualization in the next step. The final part in the process is to interprete the pattern and possibly to make a recommendation for action.

In the following, we distinguish two types of data, spatial geographic data and general spatial data.

1.2 Spatial Geographic Data

Spatial geographic data, sometimes abbreviated as geo-spatial data, distinguish themselves from general data in that associated with each object, the attributes under consideration include not only non-spatial normal attributes that also exist in other database, but also spatial attributes that are often unique or emphasized in spatial database. Spatial attributes usually describe the object's spatial information such as location and shape in the physical space.

Thus the analysis on geo-spatial data aims to extract implicit interesting knowledge such as spatial relations and patterns that are not explicitly stored in spatial databases. Such tools are crucial to organizations who make decisions based on large spatial data sets. These organizations spread across many domains including public transportation, public health, geology, resource and environmental management, agriculture, etc.

A historic spatial pattern relates to the 1855 epidemic of Asiatic cholera in London, England [44]. An epidemiologist marked all locations where the disease had struck and discovered that the locations formed a cluster whose centroid turned out to be a water-pump. When the government authorities turned off the water-pump, the cholera began to subside. Later scientists confirmed the water-borne nature of the disease.

Current approaches to spatial problems tend to use classical data mining tools after materializing the spatial relationships. Take the epidemic of cholera for example. Materializing the distances of cholera patients to the nearest water-pump would allow the classical regression tools to identify the distance to the water-pump as an important explanatory attribute. Since independent and identical distribution (iid) is usually im-

plied in classical regression models, it means the data about one patient is independent of data describing other patients. However, this is not true for spatial attributes, e.g., distance to pumps, because spatial autocorrelation states that the properties of one sample affect the properties of other samples in its neighborhood.

In this thesis, we study regression and clustering on geo-spatial data using mixture models. Regression is linked to clustering via classification. That is, clustering can be regarded as an unsupervised type of classification, which, in turn, is a specialized form of regression with the discrete target variable. The focus is on how to efficiently incorporate spatial information into the model.

1.3 General Spatial Data

Geo-spatial data become general spatial data if we no longer differentiate spatial attribute from normal attribute and treat all equally. Since every object is treated as a point in the high dimensional space, they are usually still called spatial database, as done by many researchers in spatial data mining, especially in clustering [25, 53, 100, 116, 126]. In this case, they lend themselves to classical data mining techniques that have a wide range of application, including marketing, predicting stock market and foreign exchange rate, determining commonalities and anomalies in patients, modeling proteins, finding genes in DNA sequence, etc [28].

In this thesis, on general spatial data we continue to examine clustering from another two perspectives. We concentrate on two problems, consensus clustering and outlier detection.

Like usual clustering, consensus clustering still aims to produce a good clustering for some dataset, but it operates at a higher level. It is motivated by the following examples in reality. (1) Knowledge reuse: A company wants to cluster its customers database for

marketing campaign. A variety of legacy customer segmentations have been already manually constructed based on demographics, purchasing patterns, etc. As the data size keeps increasing, the company has to employ computer techniques to automatically cluster data. However, it is reluctant to throw out all this domain knowledge, and instead wants to reuse such pre-existing knowledge to create a single consolidated clustering. (2) Distributed clustering: In practice, due to some reasons such as privacy, the whole dataset may be partitioned and allocated into different sites. For instance, every site contains all data but with a fraction of attributes, i.e., a particular view/subspace of the original data. With one subspace clustering from each site, we need to combine them to form a consolidated clustering. From above examples, we can extract the mathematical model. The input for consensus clustering is a set of partitions, rather than the original dataset as in usual clustering. The output of consensus clustering is another clustering, which is expected to be as compatible as possible with the input set.

As a complement operation to clustering, outlier detection targets those exceptional data whose pattern is rare and different from the general pattern shown by the majority of the data. It is known to all that the job of clustering is finding the general patterns/structures in the data. How about outliers, those exceptional data that cannot be put in any pigeon holes? They are usually treated as noise or error and discarded in standard clustering. It is most likely that outliers are often the results of recording error or data entry error, but they may also be legitimate data. In some situations, however, outliers bear implicit information that cannot be discovered from those canonical data. In areas like credit card fraud, telephone calling card fraud and network intrusion detection, it is those outliers that are of interest and deserve special attention. There are many definitions for outliers. Here we focus on those outliers w.r.t. both high density pattern clustering and low density pattern regularity, whose definitions will be explained

later in the thesis.

1.4 Organization of the Thesis

The rest of the thesis roughly divides into two parts based on the data type. We deal with geo-spatial data using mixture models in the first part. Chapter 2 discusses spatial regression using radial basis function networks, concentrating on incorporating spatial information by modifying model structure. Chapter 3 is devoted to spatial clustering, focusing on designing efficient Expectation-Maximization style training algorithms for Gaussian mixture. The second part handles general spatial data. Chapter 4 continues to study clustering problem at a higher level, consensus clustering, which aims to combine a given set of partitions to form a consolidated one that is most compatible with that set. Chapter 5 addresses detecting outliers. As a complement to cluster analysis, it targets the finding of those exceptional and rare data that cannot be assigned to any general pattern or cluster. Chapter 6 summarizes major results and discusses future research.

Part of this thesis has been published or accepted for publication [62, 61, 67, 64, 63, 65, 66].

Finally, it is worth noticing that all algorithms proposed in this thesis have their own limitations. They may work well on some datasets but loose to competing algorithms on other data. It is more appropriate to view them from a statistical viewpoint, which enables us to better understand different aspects of data analysis and learning:

> There is no true interpretation of anything; interpretation is a vehicle in the service of human comprehension. The values of interpretation is in enabling others to fruitfully think about an idea.
>
> -Andreas Buja

Chapter 2

SPATIAL REGRESSION USING RBF NETWORKS

2.1 Introduction

Conventional RBF networks for spatial regression assume independent and identical distribution (iid) and ignore spatial information. In this chapter, we study how to incorporate spatial content, e.g., spatial autocorrelation, into the framework of RBF networks for spatial regression.

The following is the outline of this chapter. In the rest of this section, we describe the characteristics of geo-spatial data and spatial regression problem. Then we introduce related work in Section 2.2. After reviewing RBF network for regression in Section 2.3, we present our extension of fusing data at various levels of RBF networks to incorporate spatial information in Section 2.4. Experimental evaluation is reported in Section 2.5 where we compare various fusions on real demographic datasets and investigate the effect of autocorrelation coefficient in hidden fusion. Section 2.6 concludes this chapter with a summary.

2.1.1 Geo-Spatial Data Characteristics

Geo-spatial data often exhibit two unique characteristics: spatial trend and spatial dependence [20]. Spatial trend denotes the large scale variance computed at a coarse

resolution. Spatial dependence, also called spatial autocorrelation, denotes small scale variance and has two types: positive and negative. Positive correlation means nearby sites tend to have similar characteristics and thus exhibit spatial continuity. In remote sensing images, close pixels usually belong to the same land cover type: soil, forest, etc. Negative correlation denotes nearby sites have very different characteristics.

Because of these two characteristics, iid, a fundamental assumption often made in data sampling, is no longer valid in geo-spatial data. Let us first examine independence. In practice, almost every datum is related to each other to a varying degree. For example, houses in nearby neighborhoods tend to have similar prices. This property has long ago been found by geographers who described it as the first law of geography: everything is related to everything else, but nearby things are more related than distant things [122]. As for identical assumption, there are cases of spatial data where different regions seem to have different distribution, which is referred to as spatial heterogeneity.

Let us see a real spatial dataset that clearly shows the spatial characteristics discussed above. Fig. 2.1(a) depicts crime rate information in 49 neighborhoods in Columbus Ohio, USA [6], where a site is labeled class 1 if its crime rate is higher than the mean value and labeled class 0 otherwise. We can see that in this map, most high crime sites are in the central region and low crime sites are scattered outside. Spatial trend is obvious in east-west direction, along which it shows the trend of low-high-low in crime. The data also show positive spatial autocorrelation, that is, most sites are surrounded by sites from the same class.

2.1.2 Spatial Framework

Compared to classical pattern recognition problems whose input can be usually represented by a set of feature vectors, spatial problems have an additional input, spatial framework. In this thesis, we only consider lattice data whose site index is countable

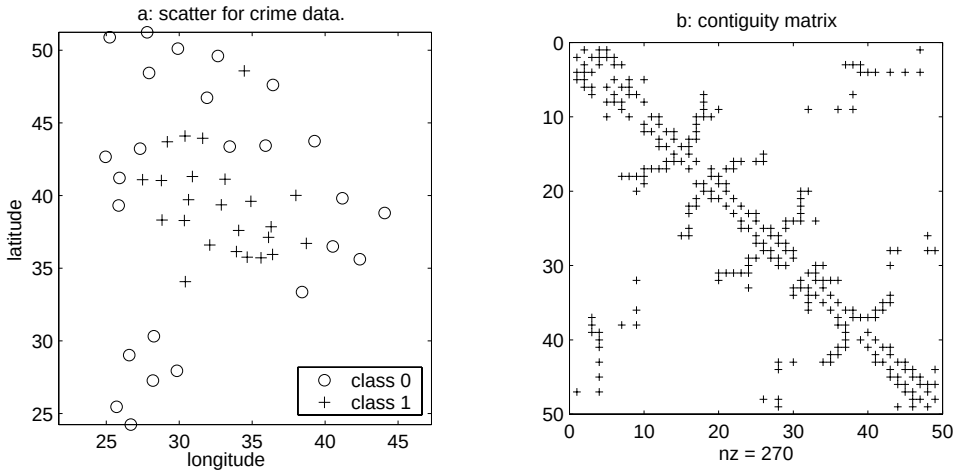

Figure 2.1: Crime rate in 49 neighborhoods (a) and its contiguity matrix (b) with a total of 270 nonzero elements $W(i,j) > 0$.

[11]. In detail, a spatial framework of n sites can be characterized by a pair (S, N), where $S = \{s_i\}_{i=1}^n$ denotes a set of n sites s_i, and $N \subseteq S \times S$ denotes the neighborhood relation. For example, S could be the set of triple (index, latitude, longitude). Two sites s_i and s_j are neighbors iff (if and only if) $(s_i, s_j) \in N, i \neq j$. For convenience, let $N(s_i) \equiv \{s_j : (s_i, s_j) \in N\}$ denote the neighborhood of s_i.

Neighborhood relation N can be given by a $n \times n$ contiguity matrix W, where $W(i,j) > 0$ iff $(s_i, s_j) \in N$ and $W(i,j) = 0$ otherwise. Although each site is actually an area, for simplicity, it is often denoted by a center point. Thus the contiguity matrix W can be computed from center points' latitude-longitude pairs. Two sites are neighbors if they are natural neighbor in Voronoi diagram (Fig. 2.2(a)) or equivalently, they are linked in the dual Delaunay triangulation (Fig. 2.2(b)). As shown in Eq. (2.1), from Voronoi diagram or Delaunay triangulation, the symmetric binary contiguity matrix W_b can be constructed, where $W_b(i,j) = 1$ iff $(s_i, s_j) \in N$ and $W_b(i,j) = 0$ otherwise. The row-normalized contiguity matrix W_n is obtained from W_b by dividing each element with the sum of its row. Consequently, W_n is also symmetric in terms of positive/zero. For example, assuming first order neighborhood, site s_1 in Fig. 2.2 has three neighbors

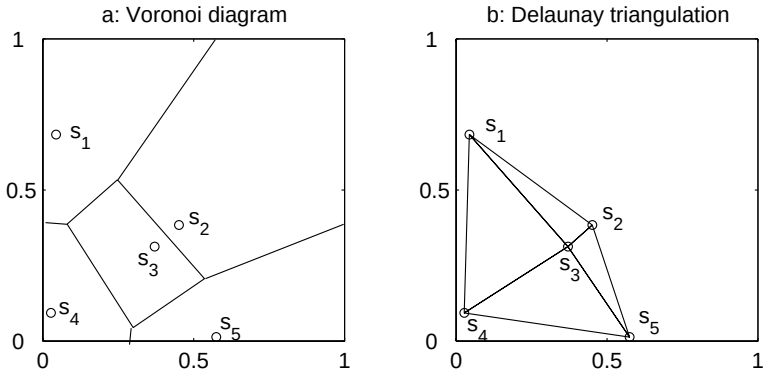

Figure 2.2: Voronoi diagram (a) and its counterpart of Delaunay triangulation (b).

s_2, s_3 and s_4, so the nonzero elements in the first row of W_b and their counterparts in W_n are $W_b(1,j) = 1$, and $W_n(1,j) = 1/3, j = 2,3,4$, respectively.

$$W_b = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix} \xrightarrow{\text{normalize}} W_n = \begin{pmatrix} 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ \frac{1}{4} & \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \end{pmatrix} \tag{2.1}$$

With neighbors defined by Voronoi diagram, the contiguity matrix of the crime data is given in Fig. 2.1(b), where a dot denotes a nonzero element. We can see that such matrices are usually sparse, that is, most of their elements are zeros. So even for a large dataset which leads to a large contiguity matrix, the storage requirement is reduced to a large extent if we only store those few nonzero elements (values and positions). Besides, some operations, like inverse, are expensive on large matrices, but there are efficient algorithms specialized for sparse matrices.

2.1.3 Problem Formulation

The problem of spatial regression can be formulated as follows:

- Given

1. A spatial framework of n sites,$S = \{s_i\}_{i=1}^{n}$. We assume that neighbor relation N is given by a row-normalized contiguity matrix W.

2. Associated with each s_i, there is a d-D feature vector of explanatory attributes $\mathbf{x}_i \equiv \mathbf{x}(s_i) \in \Re^d$ and a dependent variable $y_i \equiv y(s_i) \in \Re$ to be predicted. Let $\mathbf{y} \equiv [y_1, ..., y_n]^T$.

- Find

  A function $f : \Re^d \rightarrow \Re$. Let $\hat{y}_i \equiv f(\mathbf{x}_i)$, $\hat{\mathbf{y}} \equiv [\hat{y}_1, ..., \hat{y}_n]^T$. Here f is constrained to the model of RBF networks.

- Objective

  Maximize similarity between $\mathbf{y}$ and $\hat{\mathbf{y}}$. We use mean squared error (MSE): $\|\mathbf{y} - \hat{\mathbf{y}}\|^2/n$.

- Constraint

  Spatial autocorrelation exists,i.e., y_i is not only affected by $\mathbf{x}_i$, but also by $\mathbf{x}_j$ and y_j of its neighbors $s_j \in N(s_i)$.

2.2 Related Work

Generally speaking, current work on geo-spatial data can be divided into two fields: database and statistics. The former focuses on efficient techniques, such as storage and query, for large spatial databases [86, 26, 110, 117], and its major application includes the various geographic information systems. The latter concentrates on constructing statistical model to describe the spatial data [20, 89, 102, 121], and it is mainly applied to processing and modeling various geo-spatial data, such as demographic data and remote sensing images, etc.

Methods for incorporating spatial information roughly come in the following categories:

- Adding spatial information into dataset [71, 101, 47].
- Modifying existing algorithms, e.g., allowing an object assigned to a class iff this class already contains its neighbor [88].
- Selecting a model that encompasses spatial information [4]. This can be achieved by modifying a criterion function that includes spatial constraints [107], which mainly comes from the image analysis where Markov random field is intensively used [38].

Another category, where our approach falls, is to directly modify the structure of the model.

Compared to a lot of work in spatial contextual classification [121, 13, 59, 118], spatial regression receives less attention, not to mention application of RBF-like local expert network methods. In [40], different machine learning algorithms are applied to non-stationary spatial data analysis: using spatial coordinates to predict the rainfall. Local models, like local version of support vector regression and mixture of experts, which take into account local variability of the data (spatial heterogeneity), are found to be better than their global counterparts which are trained globally on the whole dataset. In [91], RBF coupled map lattice is used as the spatial temporal predictor to model the chaotic dynamic of radar echoes from a sea surface, and to detect embedded targets. The input is fused by weighted averaging each site and its neighbors.

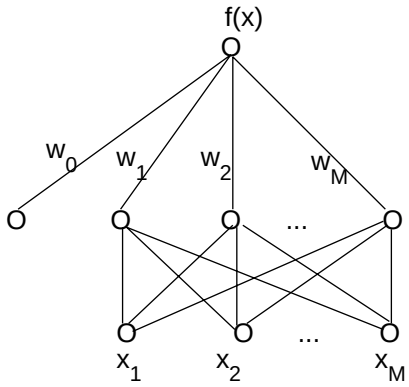

Figure 2.3: RBF network structure.

2.3 Conventional RBF Network

Conventional RBF network for regression or function approximation has been studied extensively in the literature [104, 103, 12]. It can be described mathematically as a linear combination of nonlinear radially symmetric basis functions, as shown in Eq. (2.2) and Fig. 2.3, where the basis function $\phi_m(z)$ often takes the popular Gaussian kernel in Eq. (2.3). It is proved in [55] that, given a sufficiently large number M of Gaussian kernels and the freedom to adjust center $\boldsymbol{\mu}_m$ and width h_m separately for each kernel, RBF networks can achieve arbitrarily small error.

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m \left(\frac{\|\mathbf{x} - \boldsymbol{\mu}_m\|}{h_m} \right) \tag{2.2}$$

$$\phi_m(z) = \exp(-z^2) \tag{2.3}$$

In fact, the choice of basis function is less crucial compared to the number of centers M and the width h_m. M is a hyper-parameter which determines the network structure and its estimation is costly. We select M by trial and error based on a range of values determined by the cross validation. At each iteration the input vector that results in lowering the network error the most, is used to create a hidden neuron (kernel) and it is removed from the training set [19]. This efficient process is repeated until the validation

error begins increasing. Once M is determined, centers $\boldsymbol{\mu}_m$ are chosen with K-means algorithm [82].

As for width, too small width would cause underlapping and entail a large number of kernels that lead to overfitting. On the other hand, too large width would cause overlapping and cannot give satisfactory performance. We try three ways to set constant width for all kernels: (1) The average of distance to 10th nearest neighbor (in the input vector space), which is suggested in [52]. (2) The maximum distance between centers divided by $2M$, which is used in [91]. (3) The value h that, for density estimation, minimizes the MSE between the density and the approximation [120]. It has the form in Eq. (2.4), where $\sigma^2 = \text{trace}(\Sigma)/d$ and Σ is the sample covariance matrix.

$$h = \sigma n^{\frac{-1}{d+4}} \left(\frac{4}{d+2} \right)^{\frac{1}{d+4}} \tag{2.4}$$

Once the estimation of parameters for radial basis layer is finished, the remaining task of estimating output layer weights $\mathbf{w} = [w_0, ..., w_M]^T$ is essentially a linear regression problem in Eq. (2.5), where i-th row of matrix Φ is the radial basis output vector for i-th input.

$$\mathbf{y} = \Phi \mathbf{w} \tag{2.5}$$

The MSE can be written as

$$\text{MSE}(\mathbf{w}) = \frac{1}{n} (\mathbf{y} - \Phi \mathbf{w})^T (\mathbf{y} - \Phi \mathbf{w})$$

Differentiating w.r.t. $\mathbf{w}$ we get the normal equations

$$\Phi^T(\mathbf{y} - \Phi\mathbf{w}) = 0$$

If $\Phi^T\Phi$ is nonsingular, then the unique solution is given by

$$\hat{\mathbf{w}} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y} = \Phi^+\mathbf{y} \tag{2.6}$$

where Φ^+ denotes pseudo-inverse $(\Phi^T\Phi)^{-1}\Phi^T$ for clarity.

2.4 Data Fusion in RBF Network

Spatial information, spatial autocorrelation in particular, can be incorporated into RBF network at three levels: input fusion, hidden fusion and output fusion. Input fusion is tried in [91] for regular lattice data and we adapt it to irregular lattice data. Besides, we push spatial information further into RBF network by fusing the output from hidden and output layers.

2.4.1 Input Fusion

Input fusion replaces each input with the weighted average of its neighbors and feeds the new input to a conventional RBF network. In [91], the weighting coefficient for each neighbor can be computed for spatial regular lattice data. However, the data used in our experiments are measurement for irregular lattice sites (e.g., counties) where neither the number nor the relative position of neighbors is fixed. We first average all neighbors with $W\mathbf{y}$, then by treating the result $\bar{y}_i$ (i-th element of $W\mathbf{y}$) as the only virtual neighbor for each site s_i, we can compute the correlation coefficient β between y_i and $\bar{y}_i$ in Eq. (2.7). Instead of the traditional 1-0 neural network targets, correlation-generated targets have been used in the speech recognition system to achieve better performance [131]. Similarly, the new fused input vector $\dot{\mathbf{x}}$ can be constructed by fusing the original input

$\mathbf{x}_i$ with the average of its neighbors $\bar{\mathbf{x}}_i$, as shown in Eq. (2.8), where $X = [\mathbf{x}_1, ..., \mathbf{x}_n]$, $\bar{\mathbf{x}}_i$ is the i-th column of XW^T, ρ is the coefficient linking $\mathbf{x}_i$ and its virtual neighbor $\bar{\mathbf{x}}_i$ and we set $\rho = \beta$ in this case.

$$\beta = \frac{\mathrm{Cov}(y, \bar{y})}{\sigma_y \sigma_{\bar{y}}} \tag{2.7}$$

$$\dot{\mathbf{x}}_i \equiv \frac{\mathbf{x}_i + \rho \bar{\mathbf{x}}_i}{1 + \rho} \tag{2.8}$$

2.4.2 Hidden Fusion

Hidden fusion refers to incorporating spatial autocorrelation into the output Φ from hidden radial basis layer by modifying the linear combination in Eq. (2.5). We devise two modifications: hidden fusion 1 (HF1) and hidden fusion 2 (HF2). Given in Eq. (2.9), HF1 can be interpreted as y is a linear combination of the prediction by its own attributes and by its neighbors. ρ is initially set to β obtained in Eq. (2.7) and kept fixed. With $(I + \rho W)\Phi$ replacing Φ in the original regression in Eq. (2.5), HF1's least square solution is given in Eq. (2.10).

$$\begin{aligned} \mathbf{y} &= \Phi\mathbf{w} + \rho W \Phi \mathbf{w} \qquad (2.9) \\ &= [(I + \rho W)\Phi]\mathbf{w} \\ \hat{\mathbf{w}} &= [(I + \rho W)\Phi]^{+}\mathbf{y} \qquad (2.10) \end{aligned}$$

As shown in Eq. (2.11), HF2 is obtained from HF1 in Eq. (2.9) by replacing $\Phi\mathbf{w}$ on its right-hand side with $\mathbf{y}$, i.e., the prediction replaced by the true value. It can be written as a linear regression in Eq. (2.12) where $(I - \rho W)^{-1}\Phi$ plays the role of Φ in the original regression in Eq. (2.5). The corresponding least square solution is given in Eq. (2.13).

$$\mathbf{y} = \Phi\mathbf{w} + \rho W\mathbf{y} \tag{2.11}$$

$$\mathbf{y} = [(I - \rho W)^{-1}\Phi]\mathbf{w} \tag{2.12}$$

$$\hat{\mathbf{w}} = [(I - \rho W)^{-1}\Phi]^{+}\mathbf{y} \tag{2.13}$$

For datasets whose sizes are much larger than their dimensions, usually the formed hidden layer size of RBF network (i.e., the number of radial basis centers) is larger than the input layer size(i.e., data dimension), and the hidden layer actually plays a role of nonlinearly transforming the input data to a higher dimensional space. Thus hidden fusion can be regarded as autoregression performed on the projected data in the high dimensional space. Let $\hat{\mathbf{y}}_{\mathrm{r}} = \Phi\Phi^{+}\mathbf{y}$ denote the prediction by conventional RBF network, and $\hat{\mathbf{y}}_{\mathrm{f}} = \Theta\Theta^{+}\mathbf{y}$ denote the prediction by HF2, where $\Theta = (I - \rho W)^{-1}\Phi$. Then the difference in MSE between a conventional RBF network and the corresponding HF2 is given by

$$\frac{1}{n}(\|\mathbf{y} - \hat{\mathbf{y}}_{\mathrm{r}}\|^2 - \|\mathbf{y} - \hat{\mathbf{y}}_{\mathrm{f}}\|^2) = \frac{1}{n}\mathbf{y}^T(\Theta\Theta^{+} - \Phi\Phi^{+})\mathbf{y}$$

Apparently, if $\Theta\Theta^{+} - \Phi\Phi^{+}$ is positive definite, HF2 always achieves smaller MSE. For highly correlated $W\mathbf{y}$ and $\mathbf{y}$, it is possible to make $\mathbf{y}^T(\Theta\Theta^{+} - \Phi\Phi^{+})\mathbf{y}$ positive by varying ρ, as demonstrated in later experiments.

2.4.3 Output Fusion

Output fusion is just opposite input fusion. Instead of substituting the input with the weighted average of neighbors, we can train a conventional RBF network on the original input as usual and then fuse the output with the average of neighbors. It is similar to the post-processing in spatial contextual classification after pixel-wise classification is

finished. Formally, the new prediction $\dot{\hat{\mathbf{y}}}$ by output fusion is given in Eq. (2.14), where $\hat{\mathbf{y}} = \Phi\hat{\mathbf{w}}$ denotes the prediction by a conventional RBF network, $\hat{\mathbf{w}}$ is given in Eq. (2.6), and ρ is again set to β obtained in Eq. (2.7) and kept fixed.

$$\dot{\hat{\mathbf{y}}} \equiv \frac{\hat{\mathbf{y}} + \rho W\hat{\mathbf{y}}}{1+\rho} \tag{2.14}$$

The new MSE is

$$\frac{1}{n}\|\mathbf{y} - \dot{\hat{\mathbf{y}}}\|^2 = \frac{1}{n(1+\rho)^2}\|(1+\rho)\mathbf{y} - (I+\rho W)\hat{\mathbf{y}}\|^2$$

2.5 Experimental Evaluation

2.5.1 Demographic Datasets

We evaluate various fusion on three real demographic datasets, crime [6], election [102] and house price [54, 41], all available at [90]. In the crime dataset, household income and house values in 49 neighborhoods in Columbus Ohio, USA, are treated as explanatory attributes to predict crime rate, which is shown in Fig. 2.4(a). In the election dataset, income, home ownership and population with college degrees in 3107 counties are used to predict the voting rate for 1980 USA presidential election, which is shown in Fig. 2.5(a). In house price dataset, 12 attributes, such as nitric oxides concentration, crime rate, index of accessibility to radial highways, are used to predict median values of owner-occupied homes of 506 towns in Boston area, which is shown in Fig. 2.5(c). It can be seen that all of them generally show positive spatial dependence. Spatial trend is also obvious. As illustrated in the crime dataset, for instance, high crime rate sites are clustered in the central area while low crime rate sites are scattered in the surrounding areas.

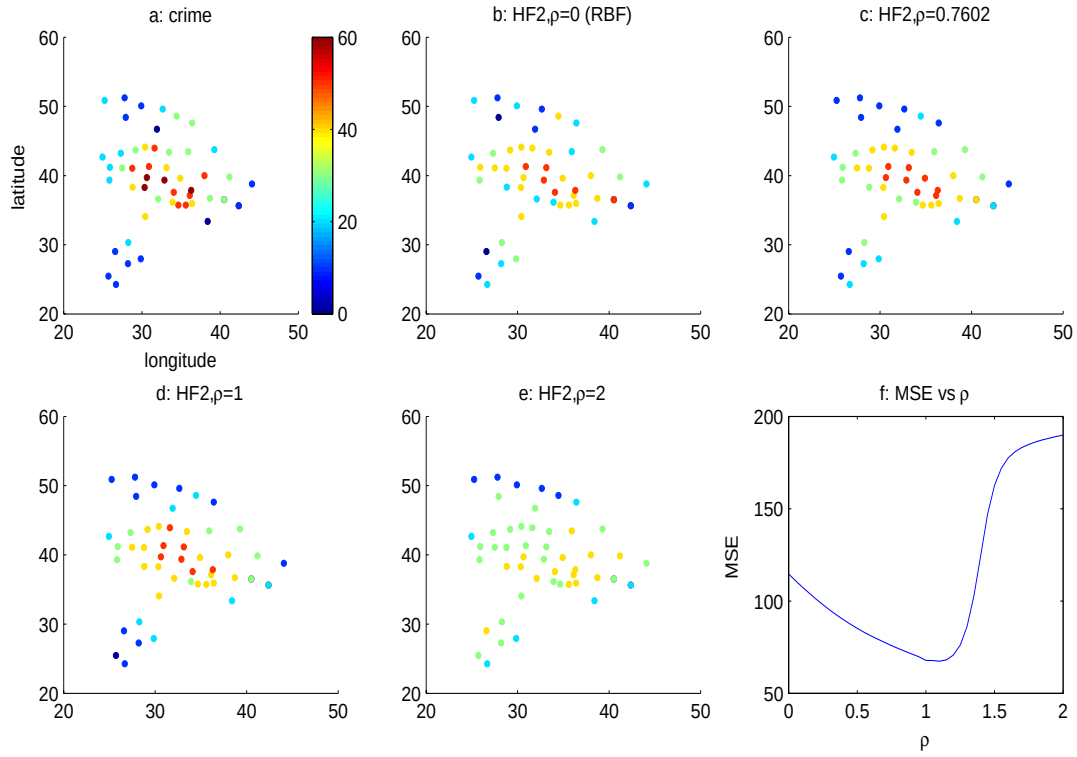

Figure 2.4: Crime data (a), its prediction (b-e) and the corresponding MSE (f) by HF2 with various ρ.

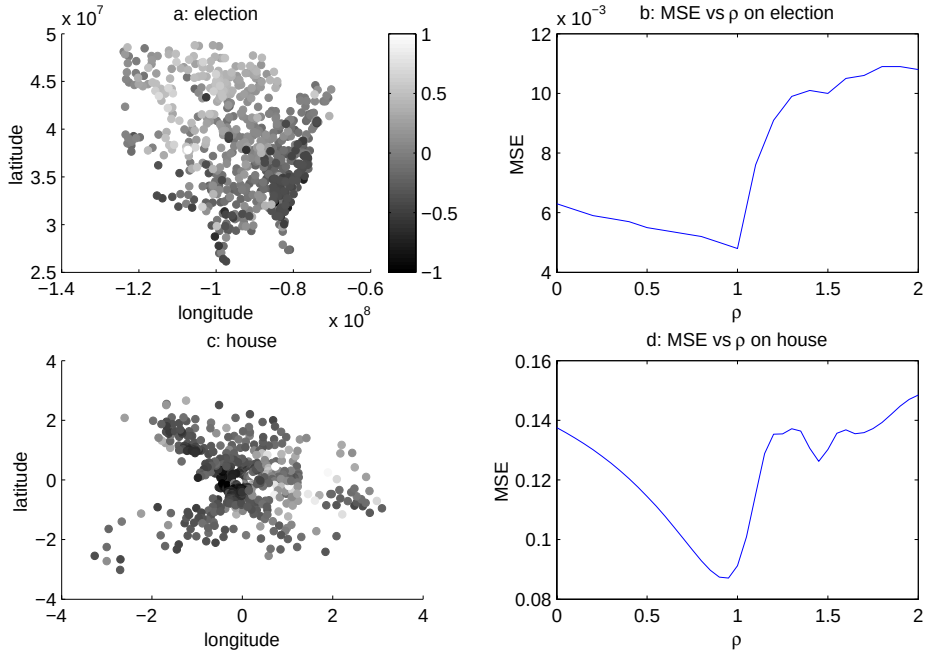

Figure 2.5: Election data (a), house price data (c), and their MSE (b,d) by HF2 with various ρ.

Table 2.1: MSE of conventional RBF network and various fusions.

	RBF	IF	HF1	HF2	OF
crime	114 ± 0.79	92 ± 2.99	92 ± 1.65	$\mathbf{84 \pm 3.71}$	105 ± 1.10
election(10^{-3})	5.7 ± 0.19	5.9 ± 0.28	5.3 ± 0.13	$\mathbf{5.1 \pm 0.08}$	5.7 ± 0.13
house(10^{-3})	142.8 ± 4.7	146.2 ± 4.6	124 ± 5.1	$\mathbf{103.3 \pm 5.1}$	135.3 ± 5.1

2.5.2 Fusion Comparison

Experiments show that the width in Eq. (2.4) always gives the best or comparable to best results, so we only report its results. The numbers of centers, 5 for crime, 100 for election and 30 for house price, are obtained with cross validation on conventional RBF networks and they are also applied in other fusions. For each dataset, there are two sets of centers, one for input fusion and the other for hidden/output fusion and conventional RBF networks.

In principle, for the test set, we must use the data for the same area but in a different year, which are unfortunately unavailable. Neither can we use cross validation by partitioning the training set into N subsets, for one site's neighbor, which is needed in various fusions, may be in another subset. Thus we can only compare various models on the same training set. For fair comparison, we generate 10 sets of centers using K-means algorithm with random initialization and early stop. The average results and their deviations are reported in Table 2.1, where RBF, IF, HF1, HF2, and OF stand for conventional RBF network, input fusion, hidden fusion 1, hidden fusion 2 and output fusion, respectively. Compared to conventional RBF networks, incorporating spatial autocorrelation by fusion at different levels generally reduces MSE with varying success. Fusing output from hidden layer gives better results than those of fusing data at two ends: raw input and final output. HF2 achieves the most significant MSE reduction on all datasets.

Table 2.2: Spatial correlation coefficient β of $\mathbf{y}$ and various $\hat{\mathbf{y}}$.

	true	RBF	IF	HF1	HF2	OF
crime	0.7602	0.5098	0.8597	0.8186	0.8789	0.8399
election	0.7575	0.6856	0.8341	0.8671	0.9308	0.9045
house	0.7778	0.3332	0.4259	0.7184	0.8829	0.7319

2.5.3 Effect of Coefficient ρ

So far, in all fusions we have set the coefficient $\rho = \beta$, the spatial autocorrelation coefficient about the true value $\mathbf{y}$. It is interesting to check the autocorrelation coefficient for various prediction $\hat{\mathbf{y}}$. The new autocorrelation is still obtained with Eq. (2.7) where $\mathbf{y}$ is replaced by $\hat{\mathbf{y}}$ and the results are listed in Table 2.2. Compared to the spatial autocorrelation of the true value, the prediction by conventional RBF networks yields lower autocorrelation. On the other hand, all fusions generally lead to higher autocorrelation in their prediction, except for the house data where only HF2 leads to higher autocorrelation.

Because the highest autocorrelation is achieved by HF2, which also achieves the lowest MSE, a natural question arises if performance of HF2 can be improved further by varying ρ in Eq. (2.11), especially by increasing it. In contrast to multi-layer feed-forward networks which require the costly error back-propagation, the major advantage of RBF networks is its quick training. In particular, the parameters of linear output layer can be solved analytically to minimize MSE, which is only feasible with a fixed ρ. Otherwise, ρ also needs to be estimated jointly with $\mathbf{w}$ using computationally expensive techniques such as Monte Carlo sampling. So it is crucial to see if we can find an optimal value for ρ.

We try a wide range $[0, 2]$ for ρ and illustrate the results in Fig. 2.4(b-f) for crime data and in Fig. 2.5(b,d) for election and house price data, respectively. Note that

when $\rho = 0$ in Eq. (2.11), HF2 is reduced to conventional RBF networks. Generally, ignoring ($\rho = 0$) and over-emphasizing ($\rho = 2$) spatial autocorrelation both lead to poor results. The former loses the spatial continuity by allowing very different sites close to one another, e.g., a few high and low crime sites are mixed together in the central area in Fig. 2.4(b). The latter usually outputs blurred result, e.g., all sites in Fig. 2.4(e) receive moderate or low values. As shown in Fig. 2.4(f) and Fig. 2.5(b,d), for all three datasets, MSE keeps decreasing as ρ grows within $[0, 1]$ and it achieves the lowest value around $\rho = 1$. Once ρ exceeds 1, MSE soon increases sharply at a larger rate than its previous decreasing rate.

Suppose that the parameters of radial basis layer are fixed and the relationship between the target y and its corresponding $(M + 1)$-D (augmented with constant 1) output vector $\boldsymbol{\phi}$ from the hidden layer is

$$y = \boldsymbol{\phi}^T \mathbf{w} + \varepsilon$$

where error $\varepsilon \sim N(0, \sigma^2)$ is independent from $\boldsymbol{\phi}$. Under this model, the least square estimates to the training data of size n are unbiased and the expected prediction error (average over everything) is approximately $\sigma^2(1+\frac{M+1}{n})$ [56]. However, this model means that y is conditionally independent given $\boldsymbol{\phi}$ (ultimately determined by the original input $\mathbf{x}$), which is invalid in the case of spatial data due to spatial constraint. A general model of spatial data is that data = trend + dependence + error [20]. Only after removing trend and dependence can we assume that the residual error is independent. Therefore it is more appropriate to describe the relationship between y and $\boldsymbol{\phi}$ with HF2's model in Eq. (2.15), where $\boldsymbol{\phi}^T \mathbf{w}$ represents spatial trend and $\rho W_y \mathbf{y}$ (W_y denotes the corresponding row in W) represents spatial dependence.

$$y = \boldsymbol{\phi}^T \mathbf{w} + \rho W_y \mathbf{y} + \varepsilon \tag{2.15}$$

2.6 Summary

Like other machine learning methods, conventional RBF networks for regression assume iid and ignore spatial information. In this chapter, we investigated various possibilities of incorporating spatial autocorrelation into RBF networks at input, hidden and output layers by fusing data belonging to the same neighborhood in the spatial space. Experiments on three real datasets show hidden fusion, HF2, always gives the best results over conventional RBF networks and other fusions. However, like total ignorance of spatial information in conventional RBF networks, over-emphasizing it also leads to poor results. Experiments suggest that the optimal value is around 1 for the coefficient ρ, which is used in HF2 to linearly combine the output from the hidden layer for each site with its neighbors.

Chapter 3

SPATIAL CLUSTERING WITH A HYBRID EM APPROACH

3.1 Introduction

Geo-spatial data often exhibit positive autocorrelation in that nearby sites tend to have similar characteristics and thus exhibit spatial continuity. In remote sensing images, close pixels usually belong to the same land cover type: soil, forest, etc. Similarly, in clustering geo-spatial data (spatial clustering for short), in addition to the object similarity in the normal attribute space, similarity in the spatial space needs to be considered and objects assigned to the same cluster should also be close to one another in the spatial space. In this chapter, using mixture models, we propose a Hybrid Expectation Maximization (HEM) approach to spatial clustering, which combines EM algorithm [21] and Neighborhood EM algorithm (NEM) [4].

The chapter outline is as follows. In the remainder of this section, we formalize the spatial clustering problem. Section 3.2 gives a literature review on related work. Basics of EM and an entropy-based view are introduced in Section 3.3, followed by NEM introduced in Section 3.4. We present our HEM approach in Section 3.5. Experimental evaluation is reported in Section 3.6 where real datasets are used for demonstration and comparison. Finally Section 3.7 concludes this chapter with a summary .

3.1.1 Problem Formulation

The goal of spatial clustering is to partition data into groups or clusters so that pairwise dissimilarity, in both attribute space and spatial space, between those assigned to the same cluster tends to be smaller than those in different clusters. Clustering is also referred to as unsupervised classification in that no prior information may be available, either on the number of clusters or what the cluster labels are. Spatial clustering can be formulated as follows:

- Given

  1. A spatial framework of n sites,$S = \{s_i\}_{i=1}^n$. We assume that neighbor relation$N$ is given by a binary contiguity matrix W whose $W(i,j) = 1$ iff $(s_i, s_j) \in N$ and $W(i,j) = 0$ otherwise.
  2. Associated with each s_i, there is a d-D feature vector of explanatory attributes $\mathbf{x}_i \equiv \mathbf{x}(s_i) \in \Re^d$.

- Find

  A many-to-one mapping $f : \{\mathbf{x}_i\}_{i=1}^n \rightarrow \{1, ..., K\}$.

- Objective

  Each object $\mathbf{x}_i$ has a true class label $y_i \in \{1, ..., K\}$. The ultimate goal is to maximize similarity between clustering and classification based on true class labels. In practice, because the class information is unavailable during learning, the objective is to optimize some criterion function such as likelihood.

- Constraint

  Spatial autocorrelation exists, i.e., $(\mathbf{x}_i, y_i)$ of site s_i may not be independent of the

corresponding values of nearby spatial sites. It is more appropriate to model the distribution of y_i as $P(y_i \mid \mathbf{x}_i, \{y_j : s_j \in N(s_i)\})$.

3.2 Related Work

Most clustering methods in the literature treat each object as a point in the high dimensional space and do not distinguish spatial attributes from normal attributes. Mainly developed in the database field, they can be divided into the following categories: partition/distance-based [82, 100], density-based [25, 5, 60], distribution-based [129], hierarchy-based [133, 45, 80], grid-based [2, 116, 126].

For spatial clustering, some methods only handle 2-D spatial attributes [27] and deal with problems like obstacles which are unique in spatial clustering [123]. Others incorporate spatial information in the clustering process, which have been reviewed in the previous chapter. Our approach HEM comes in the category of modifying a criterion function that includes spatial constraints. HEM aims to optimize the penalized likelihood, which is composed of a spatial penalty term and the likelihood, the original criterion for EM.

Clustering using mixture models with EM can be regarded as a soft K-means algorithm in that the output is posterior probability rather than hard classification. It does not account for spatial information and usually cannot give satisfactory performance on spatial data. NEM extends EM by adding a spatial penalty term in the criterion, but this makes it need more iterations in each E-step.

3.3 Basics of EM

3.3.1 Original EM

A finite mixture model of K components has the form in Eq. (3.1), where $f_k(\mathbf{x}|\theta_k)$ is k-th component's probability density function (pdf) with parameters θ_k, π_k is k-th

component's prior probability with constraint $\sum_{k=1}^{K} \pi_k = 1$ to make $f(\mathbf{x}|\Phi)$ a legal pdf. Φ denotes the set of all parameters and in the case of Gaussian mixture we use here, it includes $\{\pi_k, \boldsymbol{\mu}_k, \Sigma_k\}_{k=1}^{K}$. Given a set of data $\{\mathbf{x}_i\}_{i=1}^{n}$, the sample log likelihood function is defined in Eq. 3.2 where independence among data is implied.

$$f(\mathbf{x}|\Phi) = \sum_{k=1}^{K} \pi_k f_k(\mathbf{x}|\theta_k) \tag{3.1}$$

$$L(\Phi) = \sum_{i=1}^{n} \ln \left[\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i|\theta_k) \right] \tag{3.2}$$

In general, it is impossible to solve $\partial L/\partial \Phi = 0$ for maximum likelihood estimation. EM algorithm tries to iteratively maximize L in the context of missing data where each $\mathbf{x}$ is now augmented with a missing value $y \in \{1, ..., K\}$ indicating which component it comes from, i.e., $p(\mathbf{x}|y = k) = f_k(\mathbf{x}|\theta_k)$. It agrees with an earlier suggestion of an indirectly solvable maximum likelihood approach proposed in [23]. For Gaussian mixture problem, its convergence and advantages over other algorithms are discussed in [128]. Essentially, it produces a sequence of estimate $\{\Phi^t\}$, from an initial estimate Φ^0 and consists of two steps:

- E-step: Evaluate Q, the conditional expectation of log likelihood of the complete data $\{\mathbf{x}, y\}$ in Eq. 3.3, where $E_{\overline{P}}[\cdot]$ denotes the expectation w.r.t. the distribution $\overline{P}$ over y and in this case we set $\overline{P}(y) = P_{\Phi^{t-1}}(y) \equiv P(y|\mathbf{x}, \Phi^{t-1})$.

$$\begin{aligned} Q(\Phi, \Phi^{t-1}) &\equiv E_{\overline{P}}[\ln(P(\{\mathbf{x}, y\}|\Phi))] \\ &= E_{P_{\Phi^{t-1}}}[\ln(P(\{\mathbf{x}, y\}|\Phi))] \end{aligned} \tag{3.3}$$

- M-step: Set $\Phi^t = \text{argmax}_{\Phi} Q(\Phi, \Phi^{t-1})$. M-step can be obtained in closed form.

3.3.2 Entropy-Based View

In M-step, EM directly maximizes Q instead of L, i.e., $Q(\Phi^t, \Phi^{t-1}) \geq Q(\Phi^{t-1}, \Phi^{t-1})$. Now we prove $L(\Phi^t) \geq L(\Phi^{t-1})$ from an entropy-based viewpoint, highlighting the relationship between Q and L. Q can be written as

$$\begin{aligned}
Q(\Phi, \Phi^{t-1}) &= \sum_{i=1}^{n}\sum_{k=1}^{K} P_{\Phi^{t-1}}(y_i = k)\ln(P(\mathbf{x}_i, y_i = k|\Phi)) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} P_{\Phi^{t-1}}(y_i = k)\ln(\pi_k f_k(\mathbf{x}_i|\theta_k)) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} P_{\Phi^{t-1}}(y_i = k)\ln(f(\mathbf{x}_i|\Phi)P_\Phi(y_i = k)) \\
&= L(\Phi) - \sum_{i=1}^{n}\sum_{k=1}^{K} P_{\Phi^{t-1}}(y_i = k)\ln\left(1/P_\Phi(y_i = k)\right) \qquad (3.4) \\
&= L(\Phi) - \sum_{i=1}^{n}\sum_{k=1}^{K} P_{\Phi^{t-1}}(y_i = k)\ln\left(\frac{1}{P_{\Phi^{t-1}}(y_i = k)}\frac{P_{\Phi^{t-1}}(y_i = k)}{P_\Phi(y_i = k)}\right) \\
&= L(\Phi) - \sum_{i=1}^{n}\left[H(P_{\Phi^{t-1}}(y_i)) + D(P_{\Phi^{t-1}}(y_i)\|P_\Phi(y_i))\right] \qquad (3.5)
\end{aligned}$$

In Eq. (3.5) $H(P_{\Phi^{t-1}}(y_i))$ is the entropy of the distribution $P_{\Phi^{t-1}}(y_i)$ and $D(P_{\Phi^{t-1}}(y_i)\|P_\Phi(y_i))$ is the Kullback-Liebler distance [87] between two distributions $P_{\Phi^{t-1}}(y_i)$ and $P_\Phi(y_i)$. It is easy to show that $L(\Phi^t) \geq L(\Phi^{t-1})$ with either Eq. (3.4) or Eq. (3.5) by noting the following theorems in information and coding theory [93]. For all y_i, $\sum_{k=1}^{K} P_{\Phi^{t-1}}(y_i = k)\ln\left(1/P_\Phi(y_i = k)\right)$ on the right-hand side of Eq. (3.4), which may be called cross entropy between $P_{\Phi^{t-1}}$ and P_Φ, is minimized by setting $P_\Phi = P_{\Phi^{t-1}}$. Similarly, in Eq. (3.5), $D(P_{\Phi^{t-1}}(y_i)\|P_\Phi(y_i))$ is always non-negative. It equals zero iff $P_\Phi = P_{\Phi^{t-1}}$.

Following [98], other variants of EM such as incremental and sparse ones that partially implement E-step, can be justified in terms of a function F defined in Eq. (3.6), where $\overline{P}$ denotes a set of distributions $\{\overline{P}(y_i)\}$ and $H(\overline{P})$ denotes $\sum_{i=1}^{n} H(\overline{P}(y_i))$.

$$F(\overline{P}, \Phi) \equiv E_{\overline{P}}[\ln(P(\{\mathbf{x}, y\}|\Phi))] + H(\overline{P}) \quad (3.6)$$

$$= -D(\overline{P}\|P_\Phi) + L(\Phi) \quad (3.7)$$

By setting $\overline{P} = P_{\Phi^{t-1}}$ in Eq. (3.6) and noting that $E_{P_{\Phi^{t-1}}}[\ln(P(\{\mathbf{x}, y\}|\Phi))] = Q(\Phi, \Phi^{t-1})$, we can easily derive Eq. 3.7 from Eq. 3.5. Then EM is equivalent to the following two steps that alternately maximize F w.r.t. its two parameters, starting with an initial estimate $(\overline{P}^0, \Phi^0)$.

- E-step: Set $\overline{P}^t = \text{argmax}_{\overline{P}} F(\overline{P}, \Phi^{t-1})$. It can be shown that F is maximized by $\overline{P}^t = P_{\Phi^{t-1}}$. In that case, $F(P_{\Phi^{t-1}}, \Phi^{t-1}) = L(\Phi^{t-1})$, which is obvious from Eq. 3.7.

- M-step: Set $\Phi^t = \text{argmax}_\Phi F(\overline{P}^t, \Phi)$. It is exactly the same as M-step in EM, because $H(\overline{P})$ does not depend on Φ.

3.4 Neighborhood EM

3.4.1 Basics of NEM

To incorporate spatial information, we can add a penalty term to F that consists of $\overline{P}(y)$ for all sites. The general idea is that the penalty term will be maximized if nearby sites have similar $\overline{P}(y)$. A number of penalty terms are tried in our experiments, including sum of squared error, cross entropy, Kullback-Liebler distance, dot product. Experiments show dot product achieves the best results in terms of clustering quality and convergence. Proposed in NEM [4], it is defined in Eq. (3.8), where $\overline{P}_{ik}$ denotes$\overline{P}(y_i = k)$ and $\overline{\mathbf{P}}(y_i)$ in Eq. (3.9) denotes a column vector $[\overline{P}_{i1}, ..., \overline{P}_{iK}]$. Actually, the matrix formed by $[\overline{\mathbf{P}}(y_1), ..., \overline{\mathbf{P}}(y_n)]$ can be regarded as a fuzzy classification matrix [57]. In NEM, the new criterion to be maximized is in Eq. (3.10) where $\beta > 0$ is a fixed

coefficient.

$$G(\overline{P}) \equiv \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{K} W(i,j)\overline{P}_{ik}\overline{P}_{jk} \tag{3.8}$$

$$= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W(i,j)\overline{\mathbf{P}}(y_i)\cdot\overline{\mathbf{P}}(y_j) \tag{3.9}$$

$$U(\overline{P},\Phi) \equiv F(\overline{P},\Phi) + \beta G(\overline{P}) \tag{3.10}$$

Similar to F, U can be maximized by alternately estimating its two parameters. With $\overline{P}$ fixed, M-step can be solved analytically. In E-step where Φ is fixed, if U is maximized at $\overline{P}^*$, then $\partial U'/\partial\overline{P}_{ik} = 0$ at $\overline{P}^*$, where U' is the Lagrangian of U taking into account the constraints on $\overline{P}$. Solving it for $\overline{P}_{ik}$ yields Eq. (3.11), which can be organized as $\overline{P}^* = O(\overline{P}^*)$ to include all parameters in $\overline{P}^*$. It is proven in [4] that under certain conditions, the sequence produced by $\overline{P}^m = O(\overline{P}^{m-1})$ will converge to a fixed point to maximize U. Hence $\overline{P}^*_{ik}$ can be regarded as dot product again between the estimation from its own $\mathbf{x}$ and the estimation from its neighbors.

$$\overline{P}^*_{ik} = \frac{\pi_k f_k(\mathbf{x}_i|\theta_k)\exp\left(\beta\sum_{j=1}^{n} W(i,j)\overline{P}^*_{jk}\right)}{\sum_{l=1}^{K}\pi_l f_l(\mathbf{x}_i|\theta_l)\exp\left(\beta\sum_{j=1}^{n} W(i,j)\overline{P}^*_{jl}\right)} \tag{3.11}$$

3.4.2 Softmax Function

Let us analyze in more detail the distribution $P(y_i|N(s_i))$ provided by neighbors, which undergoes two-phase smoothing in Eq. (3.11). The first smoothing is realized by summing up over neighbors, i.e., $P(y_i = k|N(s_i)) \propto \sum_{j=1}^{n} W(i,j)\overline{P}_{jk}$. Then, to make it a legal probability, we smooth it again with softmax function, which, defined in Eq. (3.12), transfers an input vector $[p_1, ..., p_K]$ into an output vector with elements in $[0, 1]$. The resulting $P(y|N(s_i))$ through summing over neighbors and subsequent softmax transfer has the form in Eq. (3.13).

$$\text{softmax}_\beta(p_k) \equiv \frac{\exp(\beta p_k)}{\sum_{l=1}^{K} \exp(\beta p_l)} \tag{3.12}$$

$$P(y_i = k|N(s_i)) = \frac{\exp\left(\beta \sum_{j=1}^{n} W(i,j)\overline{P}_{jk}\right)}{\sum_{l=1}^{K} \exp\left(\beta \sum_{j=1}^{n} W(i,j)\overline{P}_{jl}\right)} \tag{3.13}$$

The default value of β in softmax function is one so that the vector elements' size relations are usually intact after transfer. The authors of NEM also recommend setting $\beta \in [0.5, 1]$. However, if β takes on a value greater than one, the size relations may change too, depending on the size relation of the original input. This is evident from the fact that for two positive values p_k and p_l, after transfer, their ratio becomes $\exp(\beta(p_k - p_l))$. Roughly speaking, there are two situations, as demonstrated in Fig. 3.1, where we suppose that there are four neighbors and $\sum_{k=1}^{K} p_k = 4$. As shown in Fig. 3.1(a), if $P(y_i|N(s_i))$ is very stable, that is, the mixture model fits the data quite well and there is a winner p_k much larger than all the others, setting $\beta \in [0.5, 1]$ would generally smooth $P(y_i = k|N(s_i))$ while setting $\beta > 1$ may over-emphasize the winner. On the other hand, as shown in Fig. 3.1(e), if $[p_1, ..., p_K]$ is not stable or even close to uniform, we may need to set $\beta > 1$ to magnify the impact of neighbors and strengthen the winner.

3.5 Hybrid EM

EM is not appropriate for spatial clustering because it does not account for spatial information. In contrast, although NEM incorporates spatial information, it requires more iterations in each E-step where more computation is performed to combine estimates from neighbors.

To avoid additional computation and still achieve satisfactory results on spatial data, we propose HEM, which is based on the following observation. In early passes of EM when L grows rapidly, U also grows and clustering performance increases too. U

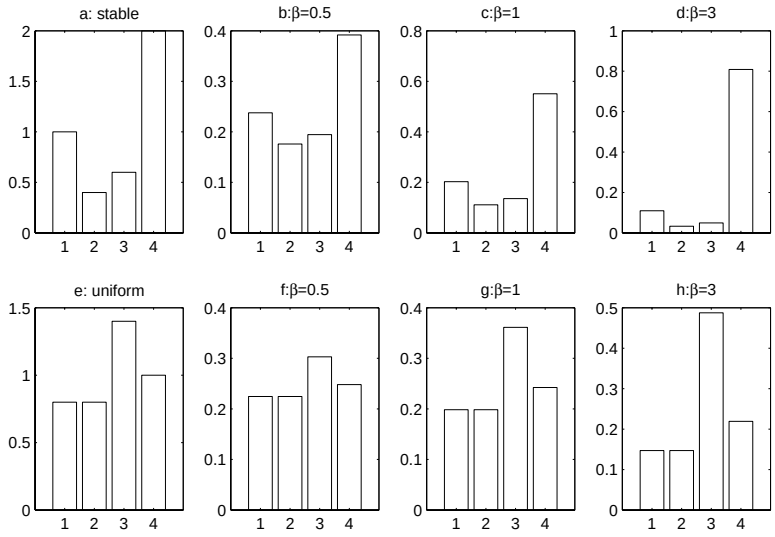

Figure 3.1: A stable input distribution (a) and its output by softmax function with different β (b-d). A uniform input distribution (e) and its output by softmax function with different β (f-h).

begins to decrease when the growth of L slows down and EM begins to converge. Such phenomenon seldom happens in NEM where clustering performance generally increases with U. This motivates us to train first using EM and turn to NEM only when U begins to decrease. Furthermore, empirical results show that we need to run E-step only once in NEM. An intuitive explanation could be that initial training with EM provides a good starting point for NEM. Such hybrid training enables our algorithm to involve much less computation than NEM and still keep U never decreasing.

We define site s_i as a *kernel site* if its largest $\overline{P}(y)$ comes from the same class as all its neighbors' do. That is, $\exists k, \forall s_j \in \{s_i\} \bigcup N(s_i), \overline{P}_{jk} = \max_l\{\overline{P}_{jl}\}$. For early training we employ a selective hard variant (winner-take-all) of EM that stands midway between K-means and EM. After E-step of EM, we transform $\overline{P}(y)$ for kernel sites into a hard distribution where all values receive zero probability except one value that is the winner (largest) in $\overline{P}(y)$. The motivation is that in spatial clustering, if spatial continuity exists, which is often the case, most sites would be surrounded by sites from the same class. Therefore, if the mixture model fits the data quite well and one site

and all its neighbors have been classified into the same class, this classification would probably be correct. Of course, such an EM variant cannot, in general, converge to the unconstrained maximum of F, even after finding Φ that maximizes F in the subsequent M-step. Nevertheless, there are computational advantages to using this variant in early training until convergence and switching to another variant that is able to find the unconstrained maximum [18]. After all, if we know which component data come from, ideally we should use data for that component only.

When such a selective hard EM cannot increase U any longer, we can fix $\overline{P}$ for kernel sites and need not to re-estimate them, since we have more confidence in the present classification of kernel sites. As demonstrated later, with proper implementation, the computation in every pass in later NEM can be saved even more by $|S_f|/n$, where S_f denotes the set of fixed sites and n is the total data size.

In detail, with pre-specified β and m (m is the number of iterations of E-step in NEM and set to 1 in our algorithm), HEM is carried out as follows with U as criterion function, starting with initial estimate $(\overline{P}^0, \Phi^0)$.

1. Selective Hard EM

   (a) E-step:

      i. Set $\overline{P}^t = \text{argmax}_{\overline{P}} F(\overline{P}, \Phi^{t-1})$, i.e., $\forall i, k$

$$\overline{P}^t_{ik} = \frac{\pi_k^{t-1} f_k(\mathbf{x}_i|\theta_k^{t-1})}{\sum_{l=1}^{K} \pi_l^{t-1} f_l(\mathbf{x}_i|\theta_l^{t-1})}$$

      ii. Transform $\overline{P}^t$ into a hard distribution for those kernel sites, i.e., for kernel site s_i, set $\overline{P}^t_{ik} = 1$ if $\overline{P}^t_{ik} = \max_l \overline{P}^t_{il}$ and set $\overline{P}^t_{ik} = 0$ otherwise.

   (b) M-step: Set $\Phi^t = \text{argmax}_{\Phi} F(\overline{P}^t, \Phi)$.

(c) Check: If $U^t \leq U^{t-1}$, go to the next step with $(\overline{P}^{t-1}, \Phi^{t-1})$, otherwise go back to E-step in EM.

2. Fix(optional)

   Fix $\overline{P}$ (binary at present) for those kernel sites S_f. We no long update $\overline{P}(y_i), s_i \in S_f$.

3. NEM

   (a) E-step: Set $\overline{P}^t = \text{argmax}_{\overline{P}} U(\overline{P}, \Phi^{t-1})$ by applying Eq. (3.11) $m = 1$ times. If fixing option is used, then apply Eq. (3.11) just for those $\overline{P}(y_i)$ whose $s_i \notin S_f$.

   (b) M-step: Set $\Phi^t = \text{argmax}_{\Phi} U(\overline{P}^t, \Phi)$. This step is exactly the same as the M-step in EM.

We have another option on when to turn. Instead of monitoring U, we can check G after E-step in EM and turn to NEM if G decreases, for G depends only on $\overline{P}$ and M-step does not change it. This would make the training turn earlier to NEM, for the increase in F may cancel the decrease in G and thus still keeps U growing. After training, $\mathbf{x}_i$ is assigned to the class k with the maximum posterior $\overline{P}_{ik}$.

3.5.1 Selective Hardening

Hardening $\overline{P}$ for those kernel sites can be justified if we decompose U as $U = \sum_{i=1}^{n} U_i(\overline{P}_i, \Phi)$ and $U_i(\overline{P}_i, \Phi)$ has the following form

$$\begin{aligned} U_i(\overline{P}_i, \Phi) &\equiv E_{\overline{P}_i}[\ln(P(\{\mathbf{x}_i, y_i\}|\Phi))] + H(\overline{P}_i) + \beta G(\overline{P}_i) \\ &= \sum_{k=1}^{K} \overline{P}_{ik} \ln(\pi_k f_k(\mathbf{x}_i|\theta_k)) + H(\overline{P}_i) + \frac{1}{2}\beta \sum_{s_j \in N(s_i)} \overline{\mathbf{P}}(y_i) \cdot \overline{\mathbf{P}}(y_j) \end{aligned}$$

Suppose that before hardening, the largest $\overline{P}(y)$ of the kernel site s_i and all its neighbors come from class k, we can derive the change in U_i after hardening as the following equation

$$\sum_{l\neq k}\overline{P}_{il}\ln(\frac{\pi_k f_k(\mathbf{x}_i|\theta_k)}{\pi_k f_k(\mathbf{x}_i|\theta_k)}) - H(\overline{P}_i) + \frac{1}{2}\beta\sum_{s_j\in N(s_i)}\sum_{l\neq k}\overline{P}_{il}(\overline{P}_{jk}-\overline{P}_{jl}) \tag{3.14}$$

If the mixture model fits the data quite well, usually $\overline{P}(y_i)$ would not be far away from $P_\Phi(y_i)$ and this implies that $P_\Phi(y_i=k)=\max_l\{P_\Phi(y_i=l)\}$, so every term in the first summation of Eq. 3.14 is positive. Apparently, the third summation is also positive. Because hard distribution's entropy is zero, the only negative term is the second term $-H(\overline{P}_i)$. Considering s_i is a kernel site, its $\overline{P}(y)$ must be quite stable, which means its $H(\overline{P}_i)$ is small. Therefore, after hardening, U_i would probably grow or at least would not decrease much.

3.5.2 Sufficient Statistics

After fixing and switching to NEM, those fixed sites' $\overline{P}(y)$ are no longer updated in E-step of NEM, so the computational complexity in E-step is proportional to $n-|S_f|$. However, if we perform M-step the usual way to update Φ,

$$\begin{aligned}
\boldsymbol{\mu}_k^t &= \frac{\sum_{i=1}^n \overline{P}_{ik}^t\mathbf{x}_i}{\sum_{i=1}^n \overline{P}_{ik}^t}\\
\Sigma_k^t &= \frac{\sum_{i=1}^n \overline{P}_{ik}^t(\mathbf{x}_i-\boldsymbol{\mu}_k^t)(\mathbf{x}_i-\boldsymbol{\mu}_k^t)^T}{\sum_{i=1}^n \overline{P}_{ik}^t}\\
&= \frac{\sum_{i=1}^n \overline{P}_{ik}^t\mathbf{x}_i{\mathbf{x}_i}^T}{\sum_{i=1}^n \overline{P}_{ik}^t} - \boldsymbol{\mu}_k^t{\boldsymbol{\mu}_k^t}^T\\
\pi_k^t &= \frac{\sum_{i=1}^n \overline{P}_{ik}^t}{n}
\end{aligned}$$

we can see that every site is still visited once. To circumvent this problem, we can use sufficient statistics. Let a vector of sufficient statistics for $(\mathbf{x}_i, y_i)$ be

$$ss_i \equiv \{\delta(y_i,k), \delta(y_i,k)\mathbf{x}_i, \delta(y_i,k)\mathbf{x}_i{\mathbf{x}_i}^T\}_{k=1}^K$$

where $\delta(y_i,k) = 1$ if $y_i = k$ and $\delta(y_i,k) = 0$ otherwise. Let $ss \equiv \sum_{i=1}^n ss_i$. The standard EM can be implemented as follows:

- E-step: Set $ss^t = E_{\overline{P}}[ss]$ with $\overline{P} = P_{\Phi^{t-1}}$. In detail, with $ss_i^t = E_{P_{\Phi^{t-1}}(y_i)}[ss_i]$, set $ss^t = \sum_{i=1}^n ss_i^t$.
- M-step: Given ss^t, set Φ^t to Φ that maximizes likelihood.

Similarly, with $\overline{P}(y)$ fixed for sites s_j in S_f and $ss_f = \sum_{s_j \in S_f} ss_j$ also fixed, the NEM part in HEM can be implemented as follows, where E-step takes time proportional to the size of sites unfixed and M-step takes constant time that is independent of data size.

- E-step: Set $ss_j^t = ss_j^{t-1}$ for $s_j \in S_f$. For $s_i \notin S_f$, set $ss_i^t = E_{\overline{P}(y_i)}[ss_i]$, where $\overline{P}(y_i)$ is obtained with Eq. 3.11. Set $ss^t = ss_f + \sum_{i \notin S_f} ss_i^t - \sum_{i \notin S_f} ss_i^{t-1}$.
- M-step: Given ss^t, set Φ^t to Φ that maximizes likelihood. In detail, suppose $ss^t = \{n_k^0, n_k^1, n_k^2\}_{k=1}^K$, then, $\forall k$,

$$\begin{aligned} \boldsymbol{\mu}_k^t &= \frac{n_k^1}{n_k^0} \\ \Sigma_k^t &= \frac{n_k^2}{n_k^0} - \boldsymbol{\mu}_k^t {\boldsymbol{\mu}_k^t}^T \\ \pi_k^t &= \frac{n_k^0}{n} \end{aligned}$$

3.6 Experimental Evaluation

3.6.1 Performance Criteria

Let us first take a look at the time complexity of the various EM-style algorithms introduced so far. Every pass consists of E-step and M-step. All have the same complexity

in M-step, $O(nK)$, except HEM with fixing, whose complexity in later NEM is reduced to $O((n - |S_f|)K)$. As for E-step complexity, EM is $O(nK)$, NEM is $O(mn^2K)$ (m is the number of iterations of E-step in NEM), HEM is $O(nK)$ in selective hard EM and $O(n^2K)$ in later NEM. The fastest is EM, closely followed by HEM, and NEM is the worst.

If every site has a true class label, although they are unavailable during training, they can be used to evaluate the final clustering quality. Let $C, Y \in \{1, ..., K\}$ denote the true class label and the cluster label, respectively. Clustering quality is measured with conditional entropy $H(C|Y)$ defined in Eq. (3.15), which can be interpreted as the remaining information in C after knowing Y. Entropy-based criteria have been successfully used in various learning systems, such as node impurity for attribute selection in decision tree [16, 105], and mutual information for discretizing input vector in hybrid speech recognition systems combining discrete hidden Markov model and neural network [108, 99]. In the extreme, it equals zero if their distributions are the same, i.e., all data from a particular class are grouped to exactly one cluster and all data in any single cluster are from the same class. We also use a more intuitive measure, error rate, which is commonly used in classification and can be regarded as a simplified conditional entropy in terms of coding. Using error rate, all data in each cluster that do not belong to the majority class of that cluster are no longer differentiated and we use one bit to encode them. For those belonging to the majority class, we assign zero bit. Therefore, error rate can be written in Eq. (3.16), where $c(k)$ denotes the majority class label in cluster k.

$$H(C|Y) = \sum_{k=1}^{K} P(Y=k)\left[\sum_{c=1}^{K} P(C=c|Y=k)\ln\left(\frac{1}{P(C=c|Y=k)}\right)\right] \quad (3.15)$$

$$E(C|Y) = \sum_{k=1}^{K} P(Y=k)\left[\sum_{c=1}^{K} P(C=c|Y=k)\times(1-\delta(c(k),c))\right] \quad (3.16)$$

3.6.2 Satimage Data

We compare HEM with EM and NEM on a real land cover dataset, Satimage, which is available at the UCI repository [97]. It consists of the four multi-spectral values of pixels in 3×3 neighborhoods in a satellite image for an area of agricultural land in Australia. The central pixel's class label from a six soil type set { red soil, cotton crop, grey soil, damp grey soil, vegetation stubble, very damp grey soil } is also provided. We only use four values for the central pixel. Because the dataset is given in random order and there is no spatial location, we synthesize their spatial coordinates by deleting the first 19 instances from the first class in the training set and allocate the remaining 4416 instances in a 64×69 grid.

4-neighborhood (up, down, left, right) is used in construction of W. The degree of spatial autocorrelation can be measured with Moran's contiguity ratio [20] for continuous attributes. For discrete attributes like soil types, we propose to use Eq. (3.17), where y denotes the true class label. In the case of regular lattice data like images, it just computes the fraction of edges shared by the pixels from the same class.

$$r = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} W(i,j)\delta(y_i,y_j)}{\sum_{i=1}^{n}\sum_{j=1}^{n} W(i,j)} \quad (3.17)$$

To emphasize spatial autocorrelation, we generate two images SAT1 and SAT2 in Fig. 3.2(a,b) with high contiguity ratio 0.9626 and 0.8858, respectively. In SAT1, all data from the same class are connected within a single block. In SAT2, each class is

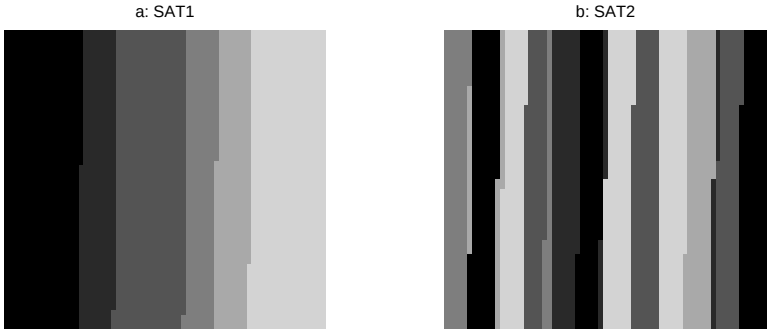

Figure 3.2: Satimage data with site's location synthesized. The contiguity ratios for (a)SAT1 and (b)SAT2 are 0.9626 and 0.8858, respectively

divided into several blocks. Within the block, data are randomly positioned.

For Gaussian mixture, we generate 10 sets of random initialization. In detail, 10 sets of centers are randomly drawn from the dataset and we partition the data into six groups based on the distance to the centers. Each component's parameters are estimated from a single group. Most of runs converge within 50 passes. To select β, we test NEM with $\beta = 0.25, 0.5, 1$. Experiments show best results are obtained with $\beta = 1$ but more iterations are needed in E-step. For SAT1, about 30/10 iterations are needed with $\beta = 1/0.5$. For SAT2, about 10/3 iterations are needed with $\beta = 1/0.5$. Table 3.1 gives the average results recorded at maximum L for EM, and maximum U for NEM and HEM, where HEM/HEMf denotes HEM without/with fixing option. For clarity, we report $-L$ and $-U$ so that all criteria in the tables are to be minimized. For comparison, we also list the results under supervised mode where each component's parameters are estimated with all data from a single class.

We can see that the entropy and error generally decrease as $-U$, rather than $-L$, decreases. Although the lowest $-L$ is achieved by EM, its entropy and error are the worst. This means that for spatial data with high spatial autocorrelation, clustering quality depends not on L, but on U which incorporates the spatial penalty term. As expected, NEM and HEM give better results on SAT1 than on SAT2, for the former's

Table 3.1: Clustering performance on Satimage data.$^{+}$SAT1 and *SAT2.

			SAT1			SAT2		
	supervised	EM	NEM	HEM	HEMf	NEM	HEM	HEMf
entropy	0.5121	0.6320	0.5391	**0.5176**	0.5276	0.5635	0.5530	**0.5520**
error	0.1508	0.2315	0.2039	**0.1919**	0.1974	0.2142	**0.2057**	0.2057
$-U(10^4)$	5.1884^{+} 5.2274^{*}	5.1406^{+} 5.1717^{*}	5.1029	**5.0807**	5.0908	5.1416	**5.1108**	5.1119
$-L(10^4)$	5.8128	**5.7711**	5.8207	5.7945	5.7974	5.8141	5.7822	5.7823

contiguity ratio is higher and hence fits our assumption more.

HEM without fixing slightly beats HEM with fixing on both datasets, probably because (1) we cannot guarantee that all kernel sites in the fixing set receive right classification, and (2) with some fixed sites, NEM cannot perform unconstrained search as it does originally. So the advantage of HEM with fixing in this case seems to be the computational cost it saves, for 48%/37% sites are fixed on the turn to NEM for SAT1/SAT2, which means that in the later NEM part, every pass needs about half computation as its counterpart does in HEM without fixing.

For SAT1/SAT2, HEM makes the switch to NEM after about 24/26 passes and slightly outperforms standard NEM in terms of all criteria after convergence. Relatively, the lead is more evident on U than on entropy and error, because of the different form of posterior they use. For many $\overline{P}(y)$, U uses their original soft forms that are different between HEM and NEM. After hardening, however, the binary forms, which are used by entropy and error, become the same. Two typical runs are depicted in Fig. 3.3(a-c) for SAT1, and in Fig. 3.3(d-f) for SAT2. The figures show that NEM initially converges faster than HEM, because NEM directly minimizes $-U$ while HEM minimizes $-F$. However, this faster speed comes with a cost, for NEM needs about 30/10 times computation in every pass for SAT1/SAT2 as HEM does. If fixing option

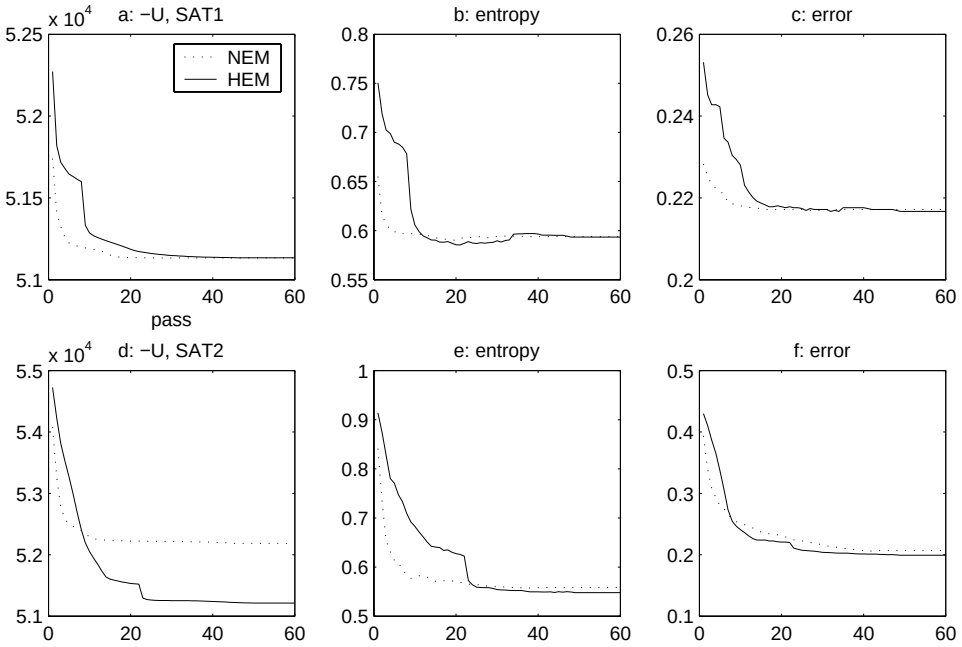

Figure 3.3: Two runs for Satimage data. (a-c) for SAT1 and (d-f) for SAT2.

is used in HEM, then after switching, this ratio nearly doubles. After about 30 passes, HEM generally catches up with NEM and converges later to a better or close solution to NEM.

To see if one iteration of E-step of NEM is really enough in HEM, we perform a series of experiments by varying the number of iterations of E-step of NEM. The average results of 10 runs are shown in Table 3.2. Note that 30/10 is the number of iterations of E-step we used in standard NEM. Although the computational cost has been increased by an order of magnitude, we can see that the improvement is not significant, especially in error rate and U.

3.6.3 House Price Data

We also evaluate HEM on the Boston house price dataset, which has been used for regression in the previous chapter. To cluster the dataset, we use 12 explanatory variables, such as nitric oxides concentration, crime rate, index of accessibility to radial highways, average number of rooms per dwelling. The clustering performance is evaluated with

Table 3.2: Clustering performance on Satimage data by HEM with varying number of iterations of E-step.

	SAT1				SAT2		
#E-step	1	10	20	30	1	5	10
entropy	0.5176	0.5095	0.5089	**0.5087**	0.5530	0.5472	**0.5468**
error	0.1919	0.1869	0.1868	**0.1867**	0.2057	0.2032	**0.2028**
$-U(10^4)$	5.0807	5.0746	5.0730	**5.0727**	5.1108	5.1091	**5.1091**
$-L(10^4)$	**5.7945**	5.7976	5.7990	5.7994	**5.7822**	5.7830	5.7830

the target variable, median values of owner-occupied homes, which is expected to have a small spread in each cluster. The house values of 506 towns in Boston area are shown again in Fig. 3.4(a). Their histogram is plotted in Fig. 3.4(b), which we can roughly model with a mixture of two components.

Using Gaussian mixture of two components, we evaluate β at 0.5,1,2, and finally set it to 1. 20 iterations are needed by E-step of NEM. Because the target variable is continuous, we cannot apply Eq. (3.15, 3.16) to compute conditional entropy or error rate and we only report $-U$ and $-L$. The average results of 10 runs are given in Table 3.3. One can see that NEM performance is slightly worse than EM in terms of U. But HEM still gives the best result. Two sample clustering results are shown in Fig. 3.4(c,d) for NEM and HEM, respectively. We can see that HEM yields a clustering with even stronger spatial continuity than that of HEM, which is also confirmed by its average U value. For this data, HEM makes the turn to NEM after about 7 passes. Although 75% sites are fixed in HEM with fixing, it leads to the same result as that without fixing. We also test HEM with different number of iterations of E-step, such as 5,10,15,20. All of them lead to results very close to standard HEM, i.e., with one iteration of E-step.

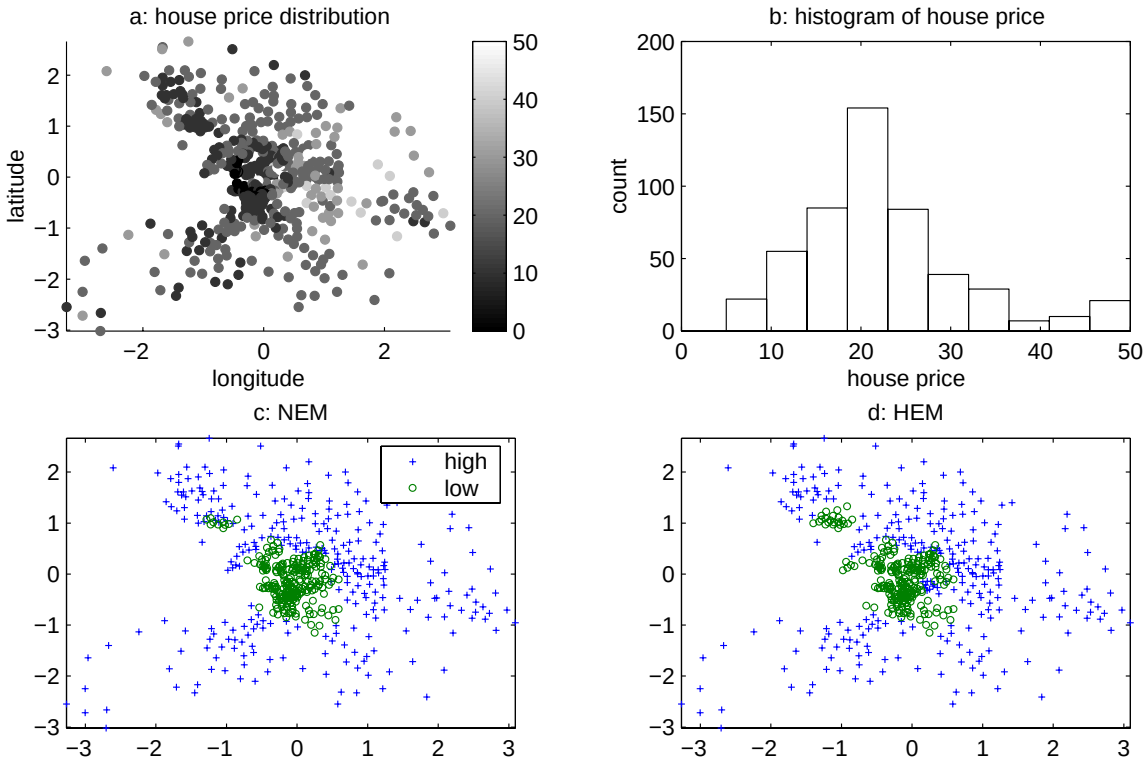

Figure 3.4: (a) shows house price distribution in 506 towns in Boston area. The corresponding histogram is plotted in (b). Two sample clustering results are shown in (c,d) for NEM and HEM, respectively.

Table 3.3: Clustering performance on house price data.

	EM	NEM	HEM
$-U(10^4)$	1.2580	1.2675	**1.2572**
$-L(10^4)$	**1.3942**	1.4014	1.3946

3.6.4 Bacteria Image

Finally, we compare HEM and NEM on an image segmentation problem to extract bacteria from background. In detail, as shown in Fig. 3.5(a), an extracted bacteria image of 40×40 is to be divided into four regions: dark region of the bacterium itself, bright region immediately surrounding the bacterium, less bright region farther away from the bacterium and grey background. The left and right boundary between the bacterium and its surrounding bright region is really very fuzzy. Due to the conflicting and mixing impact from both sides, the intensity of these border pixels are close to the grey background. Also note that in the right upper corner, there is a bright area, due to another bacterium in the original image.

With Gaussian mixture of four components, the best results of 10 runs are illustrated in Fig. 3.5(b-f). As shown in Fig. 3.5(b), since EM does not consider spatial information, its output is rather fragmented. In particular, it fails to smooth the bacterium border area, where most pixels are classified as less bright or grey, rather than dark or bright.

For NEM, first we test $\beta = 0.5, 1, 2$. With $\beta = 0.5$, we obtain results similar to EM, which means spatial information has not been emphasized enough. With $\beta = 1, 2$, we obtain results like Fig. 3.5(c). Although all clusters are connected ones, the bacterium border area is still misclassified as less bright. The reason is that the impact of its neighbors in the dark and bright regions is still very weak and the distribution offered by neighbors is unstable or close to uniform. As shown in Fig. 3.1(e-h), to change the winners from marginal winners to powerful winners and hence magnify the neighbors' correct impact, we need a large β. With $\beta = 3$ and 20 iterations of E-step, NEM produces the clustering in Fig. 3.5(d), where dark and bright regions successfully grow from both side of the border area and finally meet each other by completely occupying the border area.

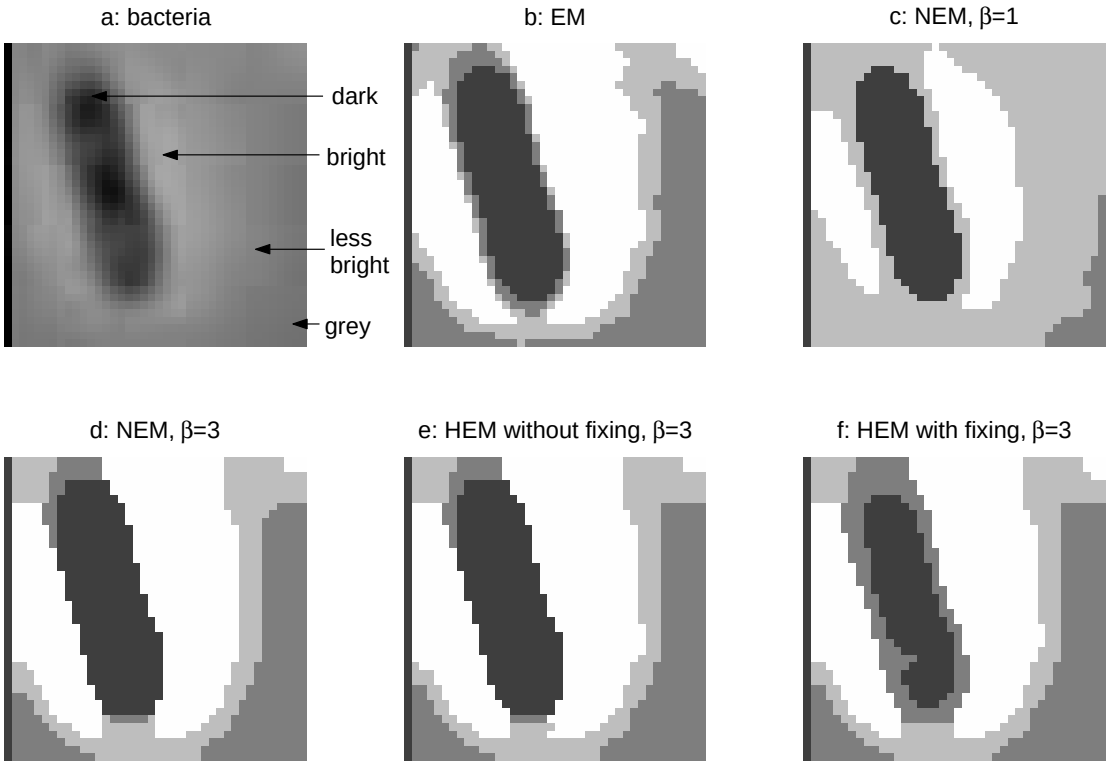

Figure 3.5: Clustering results for bacteria image. Original image (a) and various clustering results by EM (b), NEM (c-d) and HEM (e-f).

With $\beta = 3$ and no fixing, HEM generates the clustering in Fig. 3.5(e), which is very similar to NEM. Once fixing option is employed, however, HEM results in the clustering in Fig. 3.5(e) where the grey class dominates the bacterium border area, though about 60% pixels are fixed on the turn and thus 60% computation is saved in later NEM. Compared to HEM with fixing, we can see that although those border pixels are misclassified as grey on the turn in HEM without fixing, due to a large β, they are converted to dark or bright in later NEM. Detailed results are reported in Table 3.4, which indicates that HEM(without fixing) leads to a much lower $-U$ than HEMf (with fixing) does. It suggest that we should not use fixing option when the mixture model does not fit the data very well or the border area is very fuzzy.

Table 3.4: Clustering performance on bacteria image.

	EM	NEM	HEM	HEMf
$-U(10^3)$	1.238	**−0.712**	-0.705	-0.471
$-L(10^3)$	**7.325**	7.351	7.353	7.438

3.7 Summary

Spatial clustering requires consideration of spatial information and this makes EM algorithm that maximizes likelihood alone inappropriate. Although NEM algorithm incorporates a spatial penalty term, it needs much more iterations in every E-step. To incorporate spatial information while avoiding much additional computation, we proposed an HEM approach that combines EM and NEM. Early training is performed via a selective hard EM till the penalized likelihood criterion no longer increases. Then training is turned to NEM that runs only one iteration of E-step and plays a role of finer tuning. Thus spatial information is incorporated throughout HEM and the computational complexity is also retained similar to EM. Empirical results show that a few more passes are needed in HEM to converge after switching to NEM and the final clustering quality is close to or slightly better than standard NEM.

Chapter 4

CONSENSUS CLUSTERING WITH ENTROPY-BASED CRITERIA

4.1 Introduction

In this chapter, at a higher level we continue to study clustering, consensus clustering. Instead of a set of objects, the input here is a set of partitions of those objects. The goal is to produce a single consolidated partition that is as close as possible to that given set of partitions. For this purpose, two problems need to be answered. (1) How to measure distance between partitions? (2) Given a set of partitions, how to search for the consolidated one?

In the following sections we address these two problems. In detail, we first give motivation and problem formulation in the rest of this section. Related work and basics of entropy are reviewed in Sections 4.2 and 4.3, respectively. Section 4.4 gives a distribution-based view of clustering, thus paving the way for the entropy-based definition of clustering distance, which is developed in Section 4.5. Section 4.6 discusses approaches for the global optimal clustering. Section 4.7 demonstrates the properties and applications of the local optimal candidate. The combined clustering by global search methods is evaluated in Section 4.8. Finally we summarize this chapter in Sec-

tion 4.9.

4.1.1 Motivation

Given a set of N data indexed with $\{1, 2, ..., N\}$, with a prespecified number of clusters $K < N$, the aim of clustering is to assign each datum to one and exactly one cluster. The assignment can be characterized by a many-to-one mapping, $k = C(i)$, which assigns datum i to the k-th cluster. Among all these distinct clusterings, one seeks an optimal clustering C^* to achieve the required goal. Such goals can be usually quantized by a cost function such as between/within cluster scatter. Unfortunately, one cannot exhaust all possible clusterings to find the optimal one, because the number of different clusterings, $S(N, K) = \frac{1}{K!}\sum_{k=1}^{K}(-1)^{K-k}\binom{K}{k}k^N$, grows very fast [69]. For example, $S(10, 4) = 34105, S(19, 4) \approx 10^{10}$. So practical clustering algorithms only examine a very small fraction of all possible clusterings, with the goal to identify a small subset that is likely to contain the optimal, or at least the sub-optimal clustering.

Seeking more robust clusterings is the primary motivation of our work. As introduced above, clustering is a difficult problem and has been extensively studied by statistics, database and machine learning communities. Most algorithms work with numeric data [2, 45, 128, 133], but there is some work on clustering categorical data [51, 46, 37]. For clustering large data sets, some important approaches include [2, 25, 60, 133, 80, 116, 100]. The problem is challenging. High dimensionality [1, 2], data sparsity [1, 45] and noise [2, 85, 15] make clustering a harder problem. Although a number of clustering methods have been proposed, none of them are universal enough to perform equally well in all cases [134]. Differences in assumptions and contexts in different communities have made the transfer of useful generic concepts and methodologies slow to occur [72]. Since almost all clustering algorithms can only find a sub-optimal solution in practice, a natural question arises if we can obtain a better one by combining outcomes from

different clustering algorithms. Similar problems are studied extensively in multiple classifier systems, where the classifier's performance can be evaluated using the training set with known class labels. In the case of clustering, however, we have to evaluate obtained clusterings in an unsupervised way, since we don't know the true clustering.

Distributed clustering is another motivation of our work. In practice, due to some reasons such as privacy, the whole dataset may be partitioned, possibly with overlap, and each part is allocated in a different site. For example, every site contains all data but with a fraction of all attributes, that is, it stores a particular view of the original data. The clustering method has to cluster data in this subspace. This is called attribute-distributed clustering and the usefulness of having multiple views of data for better clustering is addressed in [74, 78, 94]. With one candidate clustering from each site, we need to combine them to form a consolidated one, which is expected to be better than any candidate.

4.1.2 Problem Formulation

From the motivation above, we can extract the problem formulation of consensus clustering as follows:

- Given

  A set of M candidate partitions of a common set of objects $\{x_1, ..., x_N\}$, $\Phi = \{X^m\}_{m=1}^{M}$. Let Z denote the set of natural numbers. We assume partition $X^m \in Z^N$, that is, X^m takes the form like $(1, 1, 1, 2, 2, 3, ...)$, the first three objects in cluster 1, the next two objects in cluster 2, etc.

- Find

  A combining function $f : Z^{NM} \rightarrow Z^N$, i.e., f maps each partition set Φ to another partition.

- Objective

  If there is no information about the true clustering or the relative importance of the individual candidates, then a reasonable goal is to seek one that is closest to the candidate set. If there exists a true clustering (unavailable during the combining process), naturally we hope that its distance to $f(\Phi)$ as short as possible, at least shorter than those to the candidate partitions.

- Constraint

  We are not allowed to access the original objects $\{x_1, ..., x_N\}$ or the clustering processes that produced the candidate partitions. Each object has been represented by the cluster labels assigned to it in the candidate partitions.

4.2 Related Work

4.2.1 Multiple Classifier Systems

Clustering can be regarded as an unsupervised classification problem. For its counterpart of supervised classification, there is an extensive body of work on combining multiple classifiers (or regressors) [115, 22, 39]. In fact, a related problem of multiple rankings dates back to 1785. Historical remarks are given in [95] for the theory, called relational data analysis, that relates Condorcet's solution of 1785 to the ranking problem. Among many combining techniques, boosting, in particular, has been extensively studied [114, 32, 34, 33] ever since the early 1990's. The key ideas include: (1) each classifier learns a newly weighted dataset with weight proportional to the difficulty of correctly classifying that object by previous classifiers, and (2) every classifier's performance is used to weigh its contribution to the final classification.

Because we do not know the true clustering, several problems arise in boost-clustering. One is how to assign the weight to data. Without any knowledge about the quality of

the obtained partitions, we can only assume that they are equally good and hence assign large weight to those data over which they disagree most. For instance, for any object we can find the cluster it belongs to in every partition, and measure the disagreement with the Jaccard coefficient [69], namely, the size ratio of those clusters' intersection over their union. Another problem is how to combine the candidate clusterings to form the final one. A less demanding problem is to find the best candidate among all candidates. We can choose one with the best cost function value if they are obtained by optimizing the same cost function, but it becomes less obvious otherwise. To make matters worse, some clustering algorithms, such as DBSCAN [25] and Random Walks [53], have no explicit cost functions. Under the assumption that every candidate is rather good, the best candidate could probably be the one agreed most by the whole set. So we may evaluate the degree of agreement for each candidate by measuring its average distance to all others and find the centroid candidate with the smallest distance.

4.2.2 Multi-Clustering

There is similar work on multi-clustering, constrained to the same type of clustering algorithms. That is, multiple clusterings are created and evaluated as intermediate steps in the process of attaining a single, higher quality clustering. For instance, methods are examined for iteratively improving an initial set of hierarchical clustering solutions [30]. In [29], a method is presented to obtain multiple approximate K-means solutions in main memory after making a single pass through a database. In the following we examine some recent methods in more detail.

Multi-clustering fusion methods are presented in [31, 36]. Evidence is accumulated based on combining intermediate results from an iterative clustering algorithm (e.g., K-means) with a much larger number K than the final anticipated answer. Each of K clusters of the new run is assigned to one of the previous run, resulting in a cluster

renumbering process. It enables us to update a co-occurrence matrix that records the membership degrees of data to clusters. Hence the effect of the multiple and fine-level clusterings leads to a more robust similarity indicator, which is reminiscent of the classical shared nearest neighbors measure [73]. Finally the single-link clustering is employed to recursively merges two closest clusters till some predefined criteria are met, where closeness is again based on the co-occurrence matrix. In [35], a boost-clustering algorithm is proposed to exploit the general principles of boosting. At each boosting iteration, a new training set is created using weighted random sampling from the original dataset and a simple clustering algorithm is applied to provide a new data partitioning. The final clustering solution is produced by aggregating the multiple clustering results through weighted voting.

Here consensus clustering refers to a more general problem. We just combine any given set of clusterings to produce a consolidated one without accessing the original data or any clustering algorithms that generated them. Neither do we impose any constraint on them.

4.2.3 Clustering Validity Criteria

Another related topic is clustering validity criteria. They can be classified into three categories: internal, external and relative [69]. Recent reviews are given in [49, 50, 43].

Internal criteria formulate quality as a function of the given data and/or similarities. For instance, popular evaluation criteria for compactness (within cluster scatter) include sum of squared error, which is used in standard K-means for spherical data. For separation (between cluster scatter), one can use min-cut criterion, which uses the sum of edge weights across clusters for graph partitioning. When using internal criteria, clustering becomes an optimization problem, and the clustering method can evaluate its own performance and tune its results accordingly.

On the other hand, external criteria impose quality by additional and external information, such as class labels, which is not given to the clustering methods. Considering the final judge is the human, if the true classification is known, it should be used to grade the obtained clustering. For example, by assigning all data in each cluster to the majority class of that cluster, misclassification rate can be computed against the true class labels.

Internal and external criteria are mainly based on statistical tests and their major drawback is their high computational cost. Moreover, the optimization approaches based on them aim at measuring the degree to which a data set confirms an a-priori specified scheme. As for the relative criteria, the basic idea is the evaluation of a clustering by comparing it to other clustering schemes, produced by the same algorithm but with different parameter values, such as the number of clusters K. In detail, with a suitable validity index q, for each value of K within a prespecified range, the clustering algorithm is run many times, using different set of values for the other parameters of the algorithm (e.g. different initial conditions). The best value of q obtained by each K is plotted as a function of K. We then search for a local significant change in q, which appears as a knee in the plot and it is an indication of the number of clusters underlying the dataset.

4.2.4 Distances in Clustering

Some clustering algorithms use only proximity matrices. That is, they do not access the original objects and all they need is the distance between every two objects. The computation of distance between objects usually involves the underlying distance measure for every attribute that characterizes objects. There is much work in the literature focusing on proposing or comparing such distance measures [69, 82, 42, 43]. However, little is done for comparing distinct clusterings without a common explicit cost function. Proposed for comparing true partition and the obtained clustering, Rand Index

[68] computes the fraction of all pairs of data that they agree on, that is, if the pair is assigned to the same cluster or not. Apparently, we can use one minus Rand Index as a distance measure, which equals zero iff two clusterings are identical and equals one iff two clusterings treat every pair of data differently, i.e., if the pair is assigned to the same cluster in one clustering, it must be assigned to distinct clusters in the other. To find the optimal clustering with the smallest average distance to a set of candidate clusterings, we can connect every pair of data with an edge whose weight is equal to the number of candidates that assign them to the same cluster and iteratively cut those edges with small weights. We will elaborate on this idea later in the chapter.

4.3 Basics of Entropy

Since we concentrate on clustering where each cluster can be labeled a discrete value, we only consider discrete random variables. Let X and Y be two discrete random variables that take on distinct values $x_1, ..., x_n$ and $y_1, ..., y_m$, respectively. Denoted by $H(X)$, the entropy of X defined below represents the amount of surprise, uncertainty or information in X [93, 111]. For clarity, $p(x_i) \equiv P(X = x_i)$ and it is assumed that $p\ln(1/p) = 0$ when $p = 0$. $H(X)$ is maximized when all of $p(x_i)$ are equal.

$$H(X) = \sum_{i=1}^{n} p(x_i)\ln(1/p(x_i))$$

Similarly, the entropy of joint distribution $P(X, Y)$, or joint entropy, is defined as

$$H(X, Y) = \sum_{i} \sum_{j} p(x_i, y_j)\ln(1/p(x_i, y_j))$$

with the property

$$\max\{H(X), H(Y)\} \leq H(X,Y) \leq H(X) + H(Y) \tag{4.1}$$

The average of uncertainty remaining in X after knowing Y, called conditional entropy, is defined as

$$\begin{aligned} H(X|Y) &= \sum_j p(y_j) H(X|Y = y_j) \\ &= \sum_j p(y_j) \sum_i p(x_i|y_j) \ln(1/p(x_i|y_j)) \end{aligned}$$

It can be proved that

$$H(X,Y) = H(Y) + H(X|Y) \tag{4.2}$$

which can be interpreted as the uncertainty of X and Y is equal to the uncertainty of Y plus the average uncertainty remaining in X after knowing Y. Besides, it can be shown that

$$H(X) \geq H(X|Y)$$

where the equality holds iff X and Y are independent.

4.4 Distribution-Based View of Clustering

For a particular clustering represented by X, each group of data can be labeled by a distinct value x_i the random variable X takes on. Hence we can denote the resulting n clusters by $\{x_1, ..., x_n\}$, with $P(X = x_i)$ interpreted as the fraction of data in cluster x_i. Given two clusterings $X \in \{x_i\}_{i=1}^n$ and $Y \in \{y_j\}_{j=1}^m$, we can define a new joint

Table 4.1: Two partitions X and Y.

partition	X		Y	
cluster	x_1	x_2	y_1	y_2
elements	$\{1,2\}$	$\{3,4\}$	$\{1\}$	$\{2,3,4\}$
probability	$2/4$	$2/4$	$1/4$	$3/4$

Table 4.2: Joint partition (X, Y).

joint cluster	(x_1, y_1)	(x_1, y_2)	(x_2, y_1)	(x_2, y_2)
elements	$\{1\}$	$\{2\}$	$\{\}$	$\{3,4\}$
probability	$1/4$	$1/4$	$0/4$	$2/4$

clustering, denoted by (X, Y), where each of nm clusters is uniquely labeled by a pair (x_i, y_j) and $P(X = x_i, Y = y_j)$ interpreted as the fraction of data in the intersection of clusters x_i and y_j. Similarly, conditional clustering $(Y|X)$ refers to a set of n clusterings $\{(Y|X = x_i)\}_{i=1}^{n}$, each of which partitions the data of cluster x_i into m groups according to y. Each final group, labeled $(y_j|x_i)$, consists of data in the intersection of clusters x_i and y_j. $P(Y = y_j|X = x_i)$ is interpreted as the fraction of data of cluster x_i that reside in y_j.

Let us see an example. Given a dataset of four elements $\{1, 2, 3, 4\}$ and two partitions X and Y, which are shown in Table 4.1. That is, clustering X partitions the dataset into two clusters $\{1, 2\}$ and $\{3, 4\}$. Clustering Y partitions the dataset into two clusters $\{1\}$ and $\{2, 3, 4\}$. Then the joint clustering (X, Y) partitions the dataset into four clusters, which is shown in Table 4.2. Actually there are only three clusters, because cluster (x_2, y_1) is empty. As shown in Table 4.3, $(X|Y)$ contains two conditional clusterings $(X|y_1)$ and $X|y_2)$. Note that clusters $(y_j|x_i)$ and (x_i, y_j) contain the same set of data but their probabilities $p(y_j|x_i)$ and $p(x_i, y_j)$ are different. With these distributions at hand, we can compute the corresponding entropies, such as $H(X), H(X, Y)$ and $H(Y|X)$.

We say two clusterings are independent if their respective distributions are indepen-

Table 4.3: $(Y|X)$ contains two conditional partitions $(Y|x_1)$ and $(Y|x_2)$.

conditional partition	$(Y\|x_1)$		$(Y\|x_2)$	
conditional cluster	$(y_1\|x_1)$	$(y_2\|x_1)$	$(y_1\|x_2)$	$(y_2\|x_2)$
elements	$\{1\}$	$\{2\}$	$\{\}$	$\{3,4\}$
probability	$1/2$	$1/2$	$0/2$	$2/2$

dent, i.e., $P(X,Y) = P(X)P(Y)$. For instance, clusterings X and Y defined above are not independent, because $p(x_1, y_1) \neq p(x_1)p(y_1)$. Let partition Z contain two clusters $z_1 = \{1,3\}, z_2 = \{2,4\}$, and singleton partition W contain only one cluster $w_1 = \{1,2,3,4\}$. We can see that X and Z are independent, since $\forall x_i, z_j, p(x_i, z_j) = p(x_i)p(z_j)$. X and W are also independent, since $\forall x_i, w_j, p(x_i, w_j) = p(x_i)p(w_j) = 1/2$.

4.5 Entropy-Based Clustering Distance

4.5.1 Definition

Using conditional entropy, we propose the following metric to measure distance between two clusterings X and Y on the same dataset

$$d(X,Y) \equiv H(X|Y) + H(Y|X) \tag{4.3}$$

$$= 2H(X,Y) - H(X) - H(Y) \tag{4.4}$$

where Eq. (4.4) can be derived from Eq. (4.2). Suppose X is the true clustering, then $H(X|Y)$ measures Y's within cluster scatter by computing the entropy of the distribution of each cluster of Y in X. If Y's within cluster scatter is small, each of its clusters must be contained at most in a couple of clusters of X, which means $H(X|Y)$ is small. Similarly, $H(Y|X)$ is related to Y's between cluster scatter. If clusters in Y are well separated, each of X's compact clusters must be contained at most in a couple of clusters of Y, which means $H(Y|X)$ is small.

With such a clustering distance definition, we can define the average distance from a partition X to a set of M candidate partitions $\Phi = \{X^m\}_{m=1}^M$ as

$$D(X, \Phi) \equiv \frac{1}{M} \sum_{m=1}^{M} d(X, X^m)$$

A smaller value of $D(X, \Phi)$ means a higher degree that X is agreed by Φ. When we examine partitions within this set, we can find the *local optimal/centroid clustering* X_l^*, defined as the one (within this set) that has the smallest distance, i.e.,

$$X_l^* \equiv \text{argmin}_{X \in \Phi} D(X, \Phi)$$

If this constraint is dropped, we can search for the *global optimal/centroid clustering* X_g^* over all possible clusterings X, i.e.,

$$X_g^* \equiv \text{argmin}_X D(X, \Phi)$$

Now we are able to compare different clusterings regardless of their cost functions. If the obtained candidate clusterings produced by different methods are rather good, then the quality of each clustering is inversely proportional to its average distance to all candidates. Thus we can develop a weighted version of Rand Index-based optimal clustering. That is, instead of equating the weight of edge linking two points to the number of candidates that assign them to the same cluster, it is now equated to a weighted sum. Therefore, if two points are assigned to the same cluster only by a couple of best candidates, they may still remain in the same cluster in the final clustering.

4.5.2 Properties

This symmetric distance satisfies

$$0 \leq d(X,Y) \leq H(X) + H(Y) \tag{4.5}$$

which is detailed as follows:

- It is minimized to zero iff $X = Y$.
- It is maximized to $H(X) + H(Y)$ iff X and Y are independent.
- In any other cases, the result is between 0 and $H(X) + H(Y)$.

Let us elaborate a little bit on the two extreme partitions. For a dataset of size N, denote by I_1 the singleton partition (i.e., one big cluster containing all data), and by I_N the finest partition consisting of singleton clusters (i.e., one object per cluster). For any partition X, we have $0 = H(I_1) \leq H(X) \leq H(I_N) = \ln N$ and they can be associated with the points/arcs in a set of concentric circles each of which consists of partitions with equal entropy. I_1, which is independent of any other partition X, lies in the center, since any other partition lies in the circle with radius $d(X, I_1) = H(X)$. I_N corresponds to the whole outmost circle, for $d(X, I_1) + d(X, I_N) = d(I_N, I_1) = H(I_N)$.

A *distance function* [43] must satisfy (1) $\forall X, Y, d(X,Y) = d(Y,X)$, and (2) $\forall X, d(X,X) = d_{\min}$. Based on the analysis above, we can see that clustering distance defined above is a legal distance function. In addition, to be a *metric distance function* [43], it must also fulfill (1) $\forall X, Y, d(X,Y) = 0 \Rightarrow X = Y$, and (2) the triangle inequality, that is, $\forall X, Y, Z$,

$$d(Y,Z) \leq d(X,Y) + d(X,Z) \tag{4.6}$$

Obviously, (1) is met by the clustering distance. (2) is proved in Appendix A.

Table 4.4: All five partitions for a dataset of three objects.

partition	A	B	C	D	E
clusters	$\{1\},\{2,3\}$	$\{2\},\{1,3\}$	$\{3\},\{1,2\}$	$\{1,2,3\}$	$\{1\},\{2\},\{3\}$

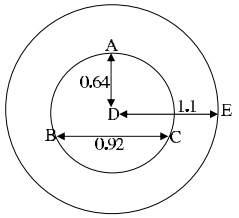

Figure 4.1: Distances among five partitions.

4.5.3 An Illustrative Example

Given a set of candidate partitions, we can find the local optimal candidate and search for the global optimal partition. With a simple dataset of three objects indexed with 1,2,3, we illustrate that the global optimum is not necessarily in that candidate set. For this dataset, there are a total of five partitions, as listed in Table 4.4. As shown in Fig. 4.1, all partitions can be visualized with two concentric circles, where partition D is located at the center, partitions A, B and C are represented by three equally spaced points at distance 0.92 in the inner circle with radius 0.64, partition E corresponds to the whole outer circle with radius 1.1 and at distance 0.46 from $A/B/C$. If the candidate set consists of A, B and E, the local optimum is E, which is also the global optimum. If C replaces E in the candidate set, however, the global optimum is still E that is no longer in the candidate set.

4.5.4 Normalized Distances

At times we need a normalized distance function with range in $[0, 1]$ and this can be obtained in several ways. The simplest one is $d_{n0}(X, Y)$ defined in Eq. (4.7) and $d_{n0}(X, Y) \leq 1$ can be proved as follows. If $H(X) + H(Y) \leq \ln N$, then from Eq. (4.5), we have $d(X, Y) \leq \ln N$ and hence $d_{n0}(X, Y) \leq 1$. If $H(X) + H(Y) > \ln N$, from Eq.

(4.4) and the fact that $H(X,Y) \leq \ln N$, we have $H(X|Y) + H(Y|X) \leq \ln N$ and hence $d_{n0}(X,Y) \leq 1$.

$$d_{n0}(X,Y) \equiv \frac{H(X|Y) + H(Y|X)}{\ln N} \tag{4.7}$$

$$d_{n1}(X,Y) \equiv \frac{1}{2}\left[\frac{H(X|Y)}{H(X)} + \frac{H(Y|X)}{H(Y)}\right] \tag{4.8}$$

$$d_{n2}(X,Y) \equiv \frac{H(X|Y) + H(Y|X)}{H(X) + H(Y)} \tag{4.9}$$

The original distance $d(X,Y)$ is upper bounded by $H(X) + H(Y)$, which generally grows as the number of clusters increases. So it may favor those with a small number of clusters. Although $d_{n0}(X,Y)$ preserves the triangle inequality, it inherits the weakness of $d(X,Y)$ and does not change relative ranking. That is, $\forall X, Y, Z, W,\ d(X,Y) > d(Z,W) \Leftrightarrow d_{n0}(X,Y) > d_{n0}(Z,W)$. We use 0 in subscript in d_{n0} to show that it is a trivial normalization. Besides, it will be far less than 1 for many pairs, for $\ln N$ is only reachable by the finest partition. It may not be a serious problem for the experiments carried out later, for all candidates have the same number of clusters prespecified equal to the number of true classes and their individual entropies may not vary much. But it is a different story if we apply different clustering algorithms that may output candidates with different number of clusters.

To make distance between a pair relatively independent of their individual entropies, two alternatives are defined in Eqs. (4.8, 4.9). For consistency, we assume that $0/0 = 0$. It is not hard to show that the former is less than or equal to the latter. Both equal 0 iff $X = Y$ and 1 iff X and Y are independent, regardless of their individual entropy sizes. However, it is unknown if triangle inequality holds for them.

With these normalized distances, the corresponding distances from a partition to a set of candidate clusterings can be similarly defined. For instance, $D_{n2}(X, \{X^m\})$

denotes such a distance based on pairwise distance d_{n2}

4.6 Toward the Global Optimum

Why stop at the local optimum? If the local optimum exhibits some desirable properties, the global optimum may possess even better properties. However, the first problem is how to search for it.

4.6.1 Simple Case

Given two clusterings X and Y, for any clustering Z, we have $d(X,Y) \leq d(Z,X) + d(Z,Y)$. Obviously, the equality holds if X and Y are independent and Z represents one big cluster containing all data. If X and Y are not independent, which is often the case, we find that the equality holds when $Z = (X,Y)$, that is, Z is the joint clustering by X and Y, because

$$\begin{aligned} & d((X,Y),X) + d((X,Y),Y) \\ = \; & [2H(X,Y) - H(X,Y) - H(X)] + [2H(X,Y) - H(X,Y) - H(Y)] \\ = \; & H(Y|X) + H(X|Y) \\ = \; & d(X,Y) \end{aligned}$$

We conclude that for two clusterings X and Y, there are at least three clusterings, X, Y and (X,Y), that have the smallest average distance to them. Similarly, such a relation can be extended to more than two clusterings, as illustrated in Figure 4.2, where, in line with Euclidean planar geometry, every point represents a clustering and a mid-point in a line segment means that its sum of distance to the two end points is equal to the distance between these two end points. For example, $d((X,Y,Z),(X,Y)) + d((X,Y,Z),Z) =$

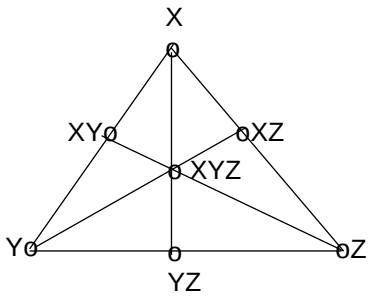

Figure 4.2: Distance relations among individual clusterings and their joint clusterings.

$d((X,Y),Z)$. Notice, however, there are other equality relations that contradict Euclidean planar geometry, such as $d((X,Y,Z),(X,Y))+d((X,Y,Z),(Y,Z)) = d((X,Y),(Y,Z))$.

Nevertheless, it becomes much more complicated when we seek a clustering with the smallest distance to a set of three candidate clusterings or more. Of course there are a great number of greedy search techniques that can be tried to yield a reasonable solution, including simulated annealing and genetic algorithms. For large datasets, however, they are impractical due to the prohibitive computational costs. Next we present two combining methods that search for a solution compatible to the candidate set in a general sense. That is, they do not explicitly check the distance to the candidate set.

4.6.2 Rand Index-Based Graph Partitioning

Rand Index considers pairwise relation of objects, that is, if two objects are assigned to the same cluster or not. Thus, for a clustering that partitions N objects, an $N \times N$ similarity matrix S can be constructed, with entry (i,j) equal to 1 when objects i and j are assigned to the same cluster, 0 otherwise. We can generalize this idea to M candidate clusterings $\Phi = \{H^m\}_{m=1}^M$. In this case, an $N \times N$ matrix S can be similarly constructed, with entry (i,j) equal to the fraction of clusterings that assign objects i and j to the same cluster. That is, $S(i,j) = \sum_{m=1}^M H^m(i,j)/M$, with $H^m(i,j) = 1$ if objects i and j are assigned together in clustering H^m, 0 otherwise.

We can see that all candidates in Φ are treated equally in computing S, since H^m will contribute 1 in the summation of $S(i,j)$ if objects i and j are assigned together in H^m. Using distance D(normalized ones), we can also develop a weighted version. First we set the weight w_m for H^m as the similarity between H^m and Φ, which is obtained with additive inversion, $w_m = 1 - D(H^m, \Phi)$. Then the weighted version is obtained with $S(i,j) = \sum_{m=1}^{M} w_m H^m(i,j)/M$.

With pairwise similarity matrix S, we can recluster objects using some reasonable similarity-based clustering method. METIS [81], a graph partitioning algorithm, is employed for its robust and scalable properties, where objects/similarities correspond to vertices/edge-weights in the graph. It tries to minimize the sum of weights of cut edges.

For example, suppose we have $M = 3$ clusterings $H^i, i = 1,2,3$, that partition $N = 8$ objects into $K = 3$ clusters, Their cluster labeling representations are given by Eq. (4.10). Then the unweighted (or equal weight) similarity matrix S is given by Eq. 4.11. Because all three distance types, $n0$, $n1$ and $n2$ yield weighted similarity matrices very similar to the unweighted, METIS produces the same cluster labeling as H^1 for all of them.

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
H^1	1	1	1	2	2	3	3	3
H^2	2	2	2	2	3	3	1	1
H^3	1	1	2	2	3	3	3	3

(4.10)

$$S = \begin{pmatrix} 1 & 1 & 0.67 & 0.33 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0.67 & 0.33 & 0 & 0 & 0 & 0 \\ 0.67 & 0.67 & 1 & 0.67 & 0 & 0 & 0 & 0 \\ 0.33 & 0.33 & 0.67 & 1 & 0.33 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.33 & 1 & 0.67 & 0.33 & 0.33 \\ 0 & 0 & 0 & 0 & 0.67 & 1 & 0.67 & 0.67 \\ 0 & 0 & 0 & 0 & 0.33 & 0.67 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0.33 & 0.67 & 1 & 1 \end{pmatrix} \tag{4.11}$$

4.6.3 Joint-Cluster Graph Partitioning

In the above method, only pairwise relation is considered and we still recluster at the resolution of the original data. Why not consider higher order relation of multiple objects?

Given $\Phi = \{H^m\}_{m=1}^M$, we have a new weighted sample that comprises $\prod_{m=1}^M |H^m|$ ($|H^m|$ denotes the number of clusters in H^m) joint-clusters in the joint clustering $\left(H^1, ..., H^M\right)$. If the candidates are similar, many joint-clusters will be empty and the sample size will be far less than $\prod_{m=1}^M |H^m|$. As stated before, every joint-cluster x can be denoted by $\left(h^1_{x_1}, ..., h^M_{x_M}\right)$, one cluster $h^m_{x_m}$ (x_m denotes the cluster label) from each candidate H^m. The weight is just the number of objects in that joint-cluster. Note that each such joint-cluster is a maximal group of objects that are completely contained in a cluster in every candidate. Since all candidates agree that all objects in the joint-cluster must stay together, we can recluster at the resolution of joint-clusters.

To use METIS, what remains is to determine similarity $S(x, y)$ between two joint-clusters $x = \left(h^1_{x_1}, ..., h^M_{x_M}\right)$ and $y = \left(h^1_{y_1}, ..., h^M_{y_M}\right)$. We propose below the cluster-wise

measure, where $|x \bigcup y|$ is the total number of objects in x and y, $|h^m_{x_m}|$ is the number of objects in cluster $h^m_{x_m}$ of candidate H^m.

For the example in Eq. (4.10), out of 3^3 joint-clusters, only six are non-empty: $jc_1 = \{v_1, v_2\}, jc_i = \{v_{i+1}\}, i = 2, ..., 5, jc_6 = \{v_7, v_8\}$. The similarity matrix for them is given by Eq. (4.12). METIS produces cluster labeling $(3, 3, 3, 2, 1, 1, 1, 1)$.

$$
\begin{aligned}
S(x,y) &\equiv \frac{1}{M}\sum_{m=1}^{M} s(h^m_{x_m}, h^m_{y_m}) \\
s(h^m_{x_m}, h^m_{y_m}) &\equiv \begin{cases} 0 & x_m \neq y_m, \text{ since } H^m \text{ does not assign them together,} \\ \frac{|x \bigcup y|}{|h^m_{x_m}|} & x_m = y_m, \text{ since } H^m \text{ assigns them together.} \end{cases}
\end{aligned}
$$

$$
S = \begin{pmatrix}
1 & 0.58 & 0.25 & 0 & 0 & 0 \\
0.58 & 1 & 0.5 & 0 & 0 & 0 \\
0.25 & 0.5 & 1 & 0.33 & 0 & 0 \\
0 & 0 & 0.33 & 1 & 0.5 & 0.25 \\
0 & 0 & 0 & 0.5 & 1 & 0.58 \\
0 & 0 & 0 & 0.25 & 0.58 & 1
\end{pmatrix} \tag{4.12}
$$

4.7 Experimental Evaluation: the Local Optimal Candidate

In this section, we demonstrate the properties and applications of the local optimal candidate with both artificial and real datasets.

4.7.1 Randomized Candidates

We devise a set of experiments to compare the true clustering and the local optimal candidate, where candidates are randomized versions of the true clustering. In detail, at

each noise level $\epsilon \in [0, 1]$, suppose we have a hypothetical true partition T with $N = 500$ data grouped into $K = 5$ clusters. Each object is labeled a random value from the uniform distribution from $1, ..., K$, but cluster sizes remain fixed at $(50, 100, 200, 50, 100)$ respectively, That is, 50 data are labeled 1, 100 data 2, etc. Then each of 10 candidate clusterings is generated by (1) randomly selecting a fraction ϵ of the data, and (2) replacing their cluster labels with random values from the uniform distribution from $1, ..., K$.

Now with T, the set Φ of 10 candidates, and the local optimal candidate X_l^* (w.r.t. Φ) at hand, the following measures are computed. First we can compute $D(X, \Phi), X = T, X_l^*$, where subscript ni $(i = 0, 1, 2)$ in D_{ni} is dropped for brevity. Note that at low noise levels, T is (close to) the global optimal partition X_g^*, so this is also a comparison of global vs. local in terms of the distance to Φ. Second, $d(X_l^*, T)$ (subscript is also dropped) is computed, which is hoped to be small in practice. For comparison, a random partition R is generated by assigning each object with a random value from the uniform distribution from $1, ..., K$. Its distances to Φ and T are also computed as a baseline.

These results are illustrated in Fig. 4.3 for 101 noise levels ϵ equally spaced in $[0, 1]$. At each level, a new set of T, Φ and R is generated. The first, second and third rows correspond to the normalized distances $n0, n1$ and $n2$, respectively. As shown in Fig. 4.3(a,d,g), $D(T, \Phi) < D(X_l^*, \Phi)$ at low noise levels, e.g., $\epsilon < 0.7$. The difference in D at very small ϵ is just the one between X_g^* and X_l^*, since $T \approx X_g^*$ then. When noise level $\epsilon > 0.7$, $D(T, \Phi) > D(X_l^*, \Phi)$, since at this time candidates in Φ have so many randomly replaced cluster labels that they share little information with T. The same reason leads to the increasing $d(T, X_l^*)$ in Fig. 4.3(b, e, h).

In both Figs. 4.3(a) and (b), the maximum distance is around 0.5. At very high noise level $\epsilon \approx 1$, all candidates in Φ are randomly generated, just the same way R is

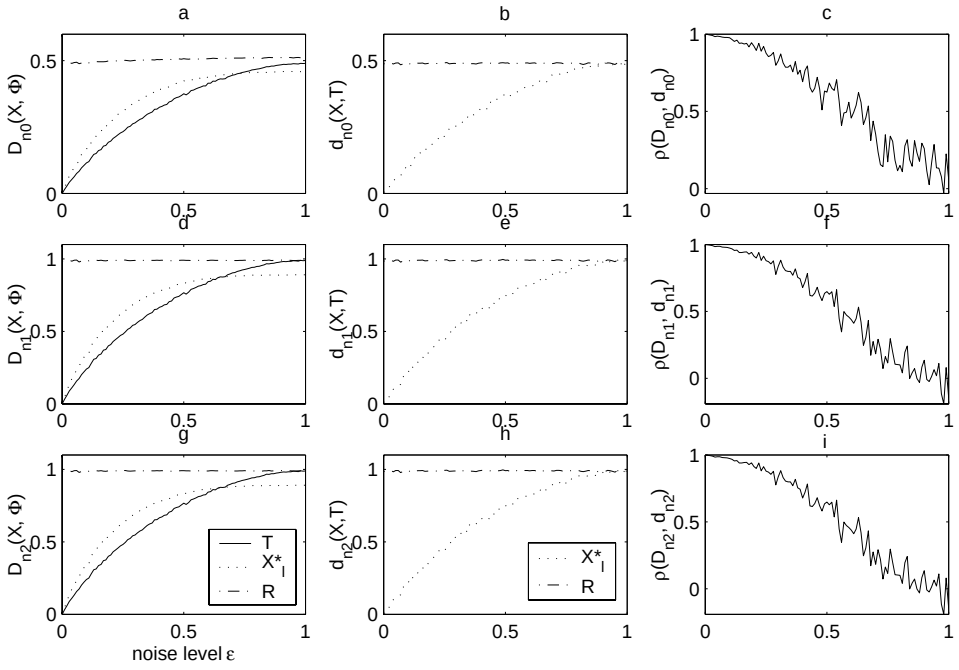

Figure 4.3: The left column shows distances to the candidate set Φ at different noise level ϵ. The corresponding distances to the true clustering T are illustrated in the middle column. The correlation coefficients ρ are plotted in the right column. From top to bottom, the three rows use distance types $n0, n1$ and $n2$, respectively.

generated. So the distance $D_{n0}(R, \Phi)$ is close to the pairwise distance $d_{n0}(R, X)$ when R and X are independent. In spite of this independence, their distance can only get as high as half one, which indicates that $\ln N$ may be too loose as the denominator in the definition of d_{n0}. On the other hand, the other two normalized distances $n1$ and $n2$ achieve a maximal value of about one at $\epsilon \approx 1$, which is desired at independence.

We claim that the local optimum from a set of good candidates is a wise choice of approximator to the true clustering. This is based on the assumption that for any clustering, its distance to that set and its distance to the true clustering are positively correlated. To show this point, in the above experiments we also compute correlation coefficient between two samples at various ϵ. In detail, at each ϵ, a new set of 100 random clusterings $\{R'_i\}_{i=1}^{100}$ are generated like R. Their distances to the candidate set Φ are stored in one sample and the corresponding distances to the true partition T

are stored in the other sample. Then the correlation coefficient $\rho(D(R', \Phi), d(R', T))$ between these two samples are computed, as illustrated in Fig. 4.3(c, f, i). One can see that positive ρ is obtained nearly at all ϵ. As expected, ρ decreases as ϵ increases, for larger ϵ means candidates in Φ show less resemblance to T and thus they are less qualified as good candidates. When $\epsilon \approx 1$, the candidates in Φ are nearly independent of T, so it is desired that $\rho(D(R', \Phi), d(R', T)) \approx 0$. We can see that ρ is still generally positive for $n0$ at this time. As for $n1$ and $n2$, it oscillates around zero, which is more desirable.

4.7.2 Candidates from the Full Space

Perhaps these entropy-based metrics can be most useful when we, without any additional knowledge, need to select a best one from a set of candidate partitions. It enables us to find the local optimum that probably will not be too bad, regardless of the data structure and the corresponding true clustering criteria. This is reminiscent of the PAC model (Probably Approximately Correct) [124] in the field of computational learning theory. Now we evaluate the local optimal candidate in the full space, that is, all candidate clusterings are obtained using all attributes.

Spherical Data

We demonstrate this assertion with 500 2-D spherical data. As shown in Fig. 4.4, they are generated with five bivariate normal distributions, 100 each, with common diagonal covariance matrix $\sigma^2 I$ and means $(0, 0)$, $(10, 0)$, $(5, 5)$, $(0, 10)$ and $(10, 10)$. In this case, Bayes classifier [23] essentially assigns data to the class with the closest mean. This is just like standard K-means algorithm [69] does with squared Euclidean distance, except that the true class means are replaced with estimated ones. Setting the number of clusters to five, we run K-means algorithm with initial centers randomly drawn from

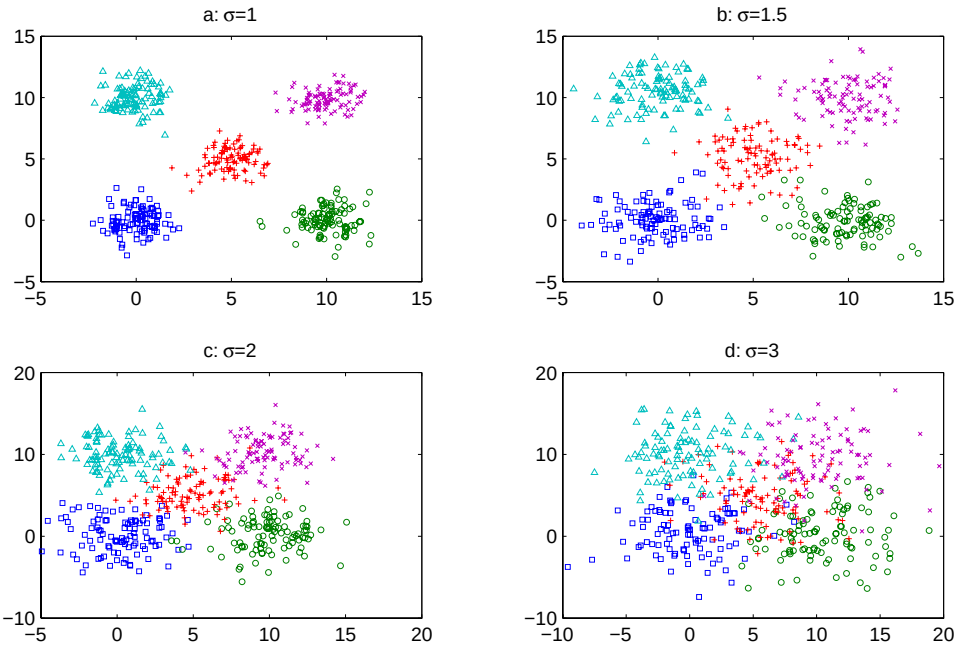

Figure 4.4: Data generated by five normal distributions with common covariance matrix $\sigma^2 I$.

data.

Similar ideas on combining multiple sets of cluster centers obtained by using K-means with different initializations appear in [14]. Here the input to our problem is sets of cluster labelings rather than class centers. To avoid the same outcomes, we iteratively run K-means algorithm until five distinct partitions are generated. Then they are ranked in terms of ascending order of distance to the true clustering. That is, if the local optimal candidate is selected with $D_{ni}(X, \Phi)$, the ranking is based on $d_{ni}(X, T)$, where X denotes the candidate, Φ the candidate set, T the true clustering.

The above experiment is repeated 100 times and the results are reported in Table 4.5. It can be seen that the local optimum will probably be top-ranked at $\sigma = 1$. This confidence declines as σ increases, for the overlap between individual classes gets more significant, as shown in Fig. 4.4(c,d). When $\sigma = 3$, the overlap is so considerable, especially for the central class, that it makes little sense to partition data into five clusters. As shown in Table 4.5, however, the heaviest frequency consistently concentrates on the

Table 4.5: Frequencies of X_l^*'s ranks on the spherical data for full space clustering.

	$n0$					$n1$					$n2$				
σ	1st	2nd	3rd	4th	5th	1st	2nd	3rd	4th	5th	1st	2nd	3rd	4th	5th
1	91	7	0	1	1	93	6	0	1	0	93	6	0	1	0
1.5	52	33	11	3	1	52	33	11	3	1	52	33	11	3	1
2	17	41	19	18	5	18	41	19	18	4	18	41	19	18	4
3	34	29	23	7	7	35	27	25	7	6	35	27	25	7	6

first or second rank for X_l^*, even at $\sigma = 3$.

Real Data

We also check these distance functions on three labeled real datasets available on the UCI repository: iris (150 data in three classes), Cleveland heart disease (303 data in two classes collected by Dr. Robert Detrano), and image segmentation (2100 data in seven classes). This time we employ EM with Gaussian mixture. After the unsupervised training, we classify data in each mixture component to the majority class of that component. For the first two datasets, original data is used. For the image data, because the error rate on the original data with EM is about 0.6, we transform them with principal component analysis [76] and only retain the first five components that contribute more than one percent of the total variance. After the transform, error rate is reduced to about 0.4.

In each experiment, we run EM with random initialization to produce five distinct partitions. Then we check the local optimum X_l^*'s rank in terms of ascending order of the distance to the true classification $D(X, T)$. This experiment is repeated 100 times and the frequencies of ranks are given by Table 4.6 for the three normalized distances $n0, n1$ and $n2$. The average error rates of fitted Gaussian mixture are about 0.1 for iris, 0.3 for heart, and 0.4 for image. Since Gaussian mixture fits the iris data very well, probably the five candidate partitions tightly center around the true classification,

Table 4.6: Frequencies of X_l^*'s ranks on the three real datasets for full space clustering.

	$n0$					$n1$					$n2$				
	1st	2nd	3rd	4th	5th	1st	2nd	3rd	4th	5th	1st	2nd	3rd	4th	5th
iris	81	10	0	4	5	81	11	0	5	3	81	11	0	5	3
heart	27	35	21	10	7	32	32	21	11	4	32	32	21	11	4
image	46	32	11	10	1	41	35	12	11	1	44	32	13	10	1

which makes the local optimum closer than others to the true classification most of times. When it comes to the heart and image data, considering the relatively high error rate and variance of the outcomes, it is hard to tell the internal relative position among the candidates. In spite of this, the sample frequency distribution is still apparently skew in that the first two ranks contribute more than half occurences.

4.7.3 Candidates from Subspaces

We have seen that the local optimal clustering is likely to be a good choice in the full space. What about attribute-distributed clustering when every candidate clustering is obtained in a subspace? In this case, the requirement of distinct candidates is dropped. Actually they are unlikely to be identical, since each on a different subspace.

For the artificial data, we simulate 500 4-D data with five Gaussian distributions, 100 each, with the common diagonal covariance matrix $0.1^2 I$, and means $(0,0,0,0)$, $(1,0,0,0)$, $(0,1,0,0)$, $(0,0,1,0)$, $(0,0,0,1)$, respectively. Four candidate clusterings are obtained with K-means in four different subspaces respectively, as shown in the first row of Table 4.7. From the four candidates, we check the ranks of the local optimal candidate X_l^* in terms of ascending order of distance to the true clustering. This experiment is repeated 100 times, each with a new dataset.

For the real data, we still use those three datasets, iris, heart and image. Again, four candidate clusterings are obtained with EM, each on a different subspace, as shown in

Table 4.7: Subspaces for candidate clusterings.

	#dim	sub 1	sub 2	sub 3	sub 4
Gaussian	4	1,2,3	2,3,4	3,4,1	4,1,2
iris	4	1,2,3	2,3,4	3,4,1	4,1,2
heart	13	1,2,3	4,5,6	7,8,9	10,11,12
image	5	1,2,3	2,3,4	3,4,5	4,5,1

Table 4.8: Frequencies of X_l^*'s ranks for subspace clustering.

	$n0$				$n1$				$n2$			
	1st	2nd	3rd	4th	1st	2nd	3rd	4th	1st	2nd	3rd	4th
Gaussian	57	25	18	0	56	26	17	1	57	25	17	1
iris	47	45	0	8	47	44	0	9	47	45	0	8
heart	87	10	1	2	33	66	1	0	25	74	1	0
image	95	4	1	0	96	3	0	1	94	4	2	0

Table 4.7. For the image data, the full space still refers to the five principal components. We repeat the experiment 100 times and record the frequencies of the local optimal candidate's ranks.

The results for the three normalized distance types $n0, n1$ and $n2$ are given in Tables 4.8, where Gaussian refers to the artificial Gaussian data. We can see that the heaviest frequency always concentrates on the first or second rank.

4.8 Experimental Evaluation: The Combined Clustering

In the following experiments, the two graph partitioning-based global search methods, Weighted Rand index-based Graph Partitioning (WRGP) and Joint-Cluster Graph Partitioning (JCGP), achieve varying success in combining candidate clusterings from either full space or subspace. Because METIS tries to produce the balanced partition (all clusters are of equal size), we only consider clustering of this type. Let us take a look at the worst time complexity for them. Suppose we have M candidate clusterings, each partitioning a set of N data into K clusters. Assuming linear complexity for graph

partitioning algorithms like METIS, then the major computation is spent in constructing similarity matrix, which is $O(KMN^2)$ for WGRI and $O(KM(K^M)^2)$ for JCHP. As we will see later, the more similar those candidates get, the fewer the non-empty joint-clusters we will have, which is the actual similarity matrix size in JCHP.

4.8.1 Randomized Candidates

First we repeat the experiment of randomized candidates with the only change that the true clustering T is balanced (each of five clusters contains 100 objects). At each noise level, WRGP and JCGP are applied to the candidate set Φ to produce a new combined clustering, whose distances to Φ and T are recorded and plotted in Fig. 4.5. For clarity, the unweighted generalized Rand Index is not shown, which is slightly worst than WRGP. WRGP with distance $n2$ is not shown either, which is very similar to that with $n1$.

We can see that both methods achieve success in this example. In terms of distance to Φ (Fig. 4.5(a,d)), at about $\epsilon < 0.7$, both methods tightly follow T, giving a smaller distance than the local optimal candidate X_l^*. When $\epsilon > 0.7$ and candidates in Φ show little resemblance to T, both methods tightly follow X_l^*, giving a smaller distance than T. In terms of distance to T (Fig. 4.5(b,d)), both methods achieve a smaller distance than X_l^*. The figures show that the difference between these two methods is not significant. A closer look indicates that at low noise levels (candidates are closer to the true clustering), JCGP slightly beats WRGP. The reverse happens at high noise levels. Table 4.9 gives some statistics over all three distance types $(n0, n1, n2)$. For instance, the second row indicates that over noise levels $[0, 0.5]$, 71% of the time JCGP's distance to the candidate set, $D(X, \Phi)$, is less than or equal to WRGP's.

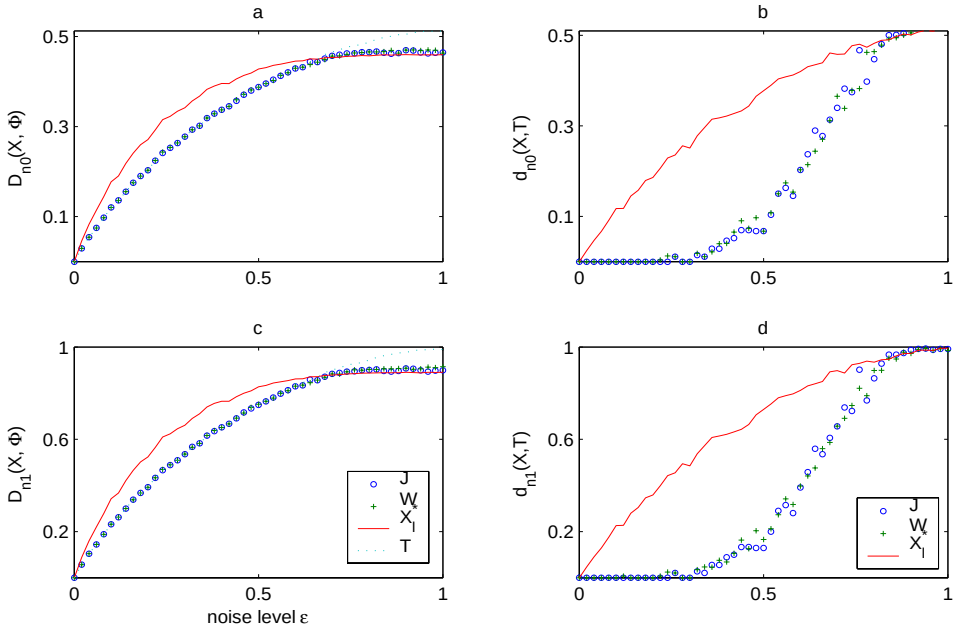

Figure 4.5: The left column shows distances to the candidate set Φ from the true clustering T, local optimal candidate X_l^*, JCGP (denoted by J) and WRGP (denoted by W) at different noise level ϵ. The corresponding distances to T from X_l^*, JCGP, and WRGP are illustrated in the right column. The top and bottom rows use distance types $n0$ and $n1$, respectively.

Table 4.9: Probabilities that HJGP yields a smaller distance than WRGP.

distance	noise level	probability
$D(X, \Phi)$	$[0, 1]$	0.64
$D(X, \Phi)$	$[0, 0.5]$	0.71
$d(X, T)$	$[0, 1]$	0.63
$d(X, T)$	$[0, 0.5]$	0.85

Table 4.10: Subspaces for candidate clusterings.

data	# dim	sub 1	sub 2	sub 3	sub 4
S1, S2	4	1,2	2,3	3,4	4,1
iris	4	1:3	2:4	3,4,1	4,1,2
heart1	13	1:3	4:6	7:9	10:12
heart2	13	1:7	3:9	5:11	7:13

4.8.2 Candidates from Subspaces

From now on, because distance types $n1$ and $n2$ always yield similar results, we only report the results of $n0$ and $n1$.

For attribute-distributed clustering, we generate 100 4-D data from each of five Gaussian distributions with the common diagonal covariance matrix $\sigma^2 I$ and means $(0,0,0,0)$, $(1,0,0,0)$, $(0,1,0,0)$, $(0,0,1,0)$, $(0,0,0,1)$, respectively. Two datasets are generated, one with $\sigma = 0.1$ and the other with $\sigma = 0.3$. We refer to the former as S1 and the latter S2. K-means can easily find the true classification in the full space for S1, but not S2. Four candidate clusterings are obtained with K-means in four different subspaces respectively, as shown in the first row of Table 4.10. For convenience, we use $i:j$ to denote $i, i+1, ..., j-1, j$. WRGP and JCGP are applied to the candidate set to produce the combined clusterings. For the real data, we use two datasets, iris and heart. Again, four candidate clusterings are obtained with EM, each on a different subspace, as shown in Table 4.10. Two sets of subspaces are tried for the heart data.

The above experiments are run 10 times and the median distance values are given in Tables 4.11 and 4.12 for JCPG, WRGP, the local optimal candidate X_l^*, and the *local worst candidate* X_l^+ (whose distance to Φ is the largest). One can see that both JCGP and WRGP perform best on S1 and S2 in terms of $d(X,T)$. The improvement on $D(X,\Phi)$ is less significant on these two datasets. For the iris data, although both JCGP and WRGP lead to a smaller $d(X,T)$ than X_l^*, they lead a higher $D(X,\Phi)$. For

Table 4.11: The median distance values for subspace clustering with distance type $n0$.

	$d(X,T)$				$D(X,\Phi)$			
	JCGP	WRGP	X_l^*	X_l^+	JCGP	WRGP	X_l^*	X_l^+
S1	0.0032	**0**	0.1939	0.2083	0.2085	**0.2054**	0.2455	0.2523
S2	0.1937	**0.1865**	0.2942	0.2932	0.2755	**0.2740**	0.2802	0.2867
iris	0.0759	**0.0639**	0.1183	0.2225	0.1121	0.1082	**0.1047**	0.1423
heart1	0.2243	0.1968	**0.1941**	0.1992	0.1768	0.1642	**0.1410**	0.1592
heart2	0.1940	0.1999	0.1941	**0.1704**	0.1101	0.1381	**0.0916**	0.1437

Table 4.12: The median distance values for subspace clustering with distance type $n1$.

	$d(X,T)$				$D(X,\Phi)$			
	JCGP	WRGP	X_l^*	X_l^+	JCGP	WRGP	X_l^*	X_l^+
S1	0.0061	**0**	0.4055	0.4239	0.4050	**0.4001**	0.4873	0.4962
S2	0.3742	**0.3525**	0.5710	0.5676	0.5349	**0.5261**	0.5477	0.5621
iris	0.1732	**0.1377**	0.2188	0.5327	0.2691	0.2590	**0.2537**	0.3523
heart1	0.9566	**0.8131**	0.8212	0.9994	0.8039	0.7394	**0.7211**	0.7442
heart2	0.9808	0.8521	0.9994	**0.7705**	0.5467	0.6715	**0.4773**	0.7120

the two heart datasets, the only improvement is that WRGP yields a smaller distance $d(X,T)$ than X_l^* with distance type $n1$ on data heart1. In general, compared to JCGP, WRGP always leads to a better or comparable result. However, as for computational complexity, the similarity matrix size is much smaller for JCGP (equal to the number of non-empty joint-clusters), which is given in Table 4.13. Note that the corresponding size for WRGP is just the data size.

Discussion

Let us explore the underlying reasons in more detail using the results with distance type $n1$ in Table 4.12, since $n1$ is less sensitive to individual entropies.

Table 4.13: The average number of joint-clusters in JCGP.

data	S1	S2	iris	heart1	heart2
# joint-clusters	50	100	12	8	16

First, with the results of the two heart datasets, we show that the quality (i.e., $d(X,T)$) of the combined clustering depends on the distance of the candidates to the true clustering. In data heart1, the majority of candidates have a smaller distance $d(X,T)$ ($d(X_l^*,T) = 0.1941$) than the minority of candidates ($d(X_l^+,T) = 0.1992$). Since both methods try to find the centroid clustering compatible to the majority of candidates, the combined clustering has a smaller distance $d(X,T)$ than X_l^+, though not necessarily than X_l^*. It is a different story when it comes to data heart2, where the majority of candidates have a larger distance $d(X,T)$ ($d(X_l^*,T) = 0.9994$) than the minority of candidates ($d(X_l^+,T) = 0.7705$). In this case, both methods lead to a clustering that has a larger distance $d(X,T)$ than X_l^+.

Second, with the results of data S1 and iris, we show that the candidate's relative position to the true clustering is another more important factor determining the quality of the combined clustering. Comparing $d(X,T)$ of the candidates for these two datasets, one can see that S1's candidates are not better than those of iris, especially for X_l^*. However, great success is achieved by both methods on S1, rather than on iris. Why? Because S1 provides the ideal situation for the combining methods, i.e., all attributes are independent from one another and their contribution to the clustering in the full space is also independent. In this case, as shown in Fig. 4.6(a), the candidates from different subspaces would evenly center around the true clustering and they are complementary to one another in the combining process. S1's four candidate clusterings are shown in Figs. 4.7(a-d), where they are projected to the first two principal components directions obtained from the full space. Note that each cluster of candidates is marked with the marker of the corresponding true cluster in the full space with which it shares most objects. It is possible that more than one cluster of the candidate are mapped to the same true cluster, e.g., only four markers are used for five clusters in Fig. 4.7(a). In each

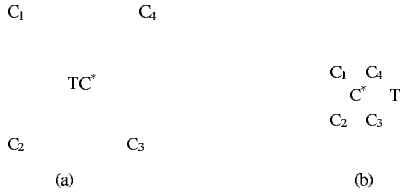

Figure 4.6: Both (a) and (b) show a true clustering T, and a set of four candidate clusterings $\{C_1, C_2, C_3, C_4\}$ for which C^* is the centroid. Although the average distance to T is larger for candidates in (a) than those in (b), their centroid C^* is closer to T than the counterpart in (b).

Table 4.14: The median distance values for full space clustering with distance type $n0$.

	$d(X,T)$				$D(X,\Phi)$			
	JCGP	WRGP	X_l^*	X_l^+	JCGP	WRGP	X_l^*	X_l^+
S2	**0.1272**	0.1291	0.1780	0.1949	**0.0841**	0.0896	0.0897	0.1455
iris	0.0764	**0.0742**	0.0864	0.0965	0.0330	0.0562	**0.0276**	0.0924
heart	0.1847	**0.1824**	0.1825	0.1941	0.0885	0.0870	**0.0853**	0.1499

subspace a different subset of original clusters can be correctly identified. As shown in Fig. 4.7(e,f), combining them by either method gives the exact true clustering, which is also obtainable by K-means in the full space.

4.8.3 Candidates from the Full Space

We also evaluate the combining methods when candidate are from the full space. We don't use Gaussian data S1, because it is too easy for K-means to find the true clustering in the full space. We use the other three datasets, Gaussian data S2, iris and heart. K-means/EM is used on data S2/(iris,heart) to generate a set of 10 candidate clusterings, to which JCGP and WRGP are applied to produce a combined new clustering. This experiment is repeated 10 times and the median distance values are given by Tables 4.14 and 4.15.

In terms of $D(X,\Phi)$, both methods lead to a smaller distance than X_l^* only on data S2. They fail on data iris and heart. In terms of $d(X,T)$ that is our ultimate goal, both methods succeed on data S2 and iris. On data heart, only WRGP leads to a slightly

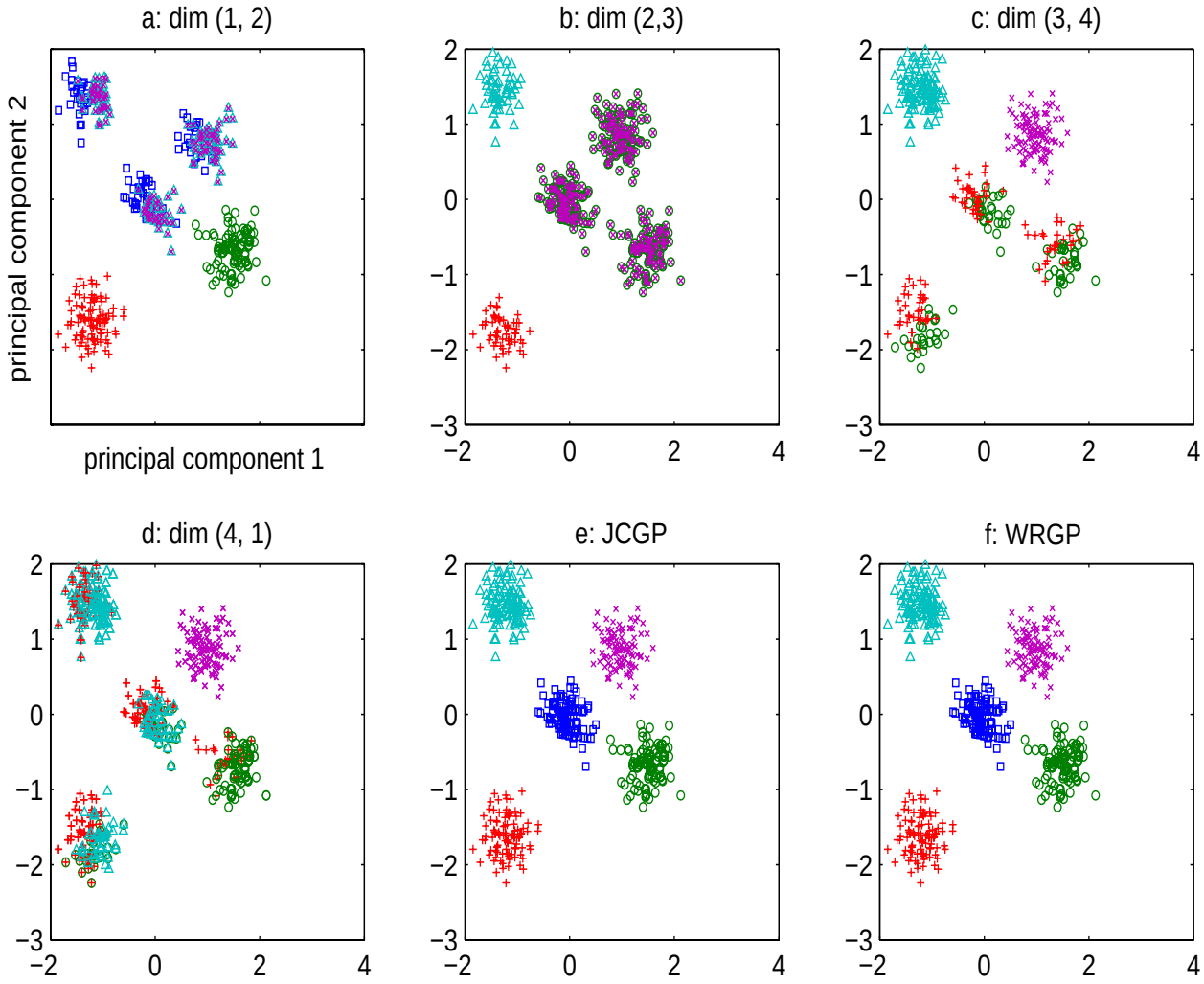

Figure 4.7: Four candidate clusterings (a-d) are from four subspaces. They are plotted in the space of the first two principal components obtained from the full space. Both JCGP (e) and WRGP (f) give the true clustering.

Table 4.15: The median distance values for full space clustering with distance type $n1$.

	$d(X,T)$				$D(X,\Phi)$			
	JCGP	WRGP	X_l^*	X_l^+	JCGP	WRGP	X_l^*	X_l^+
S2	**0.2457**	0.2479	0.2880	0.3808	**0.1639**	0.1726	0.1989	0.2864
iris	0.1775	**0.1692**	0.1975	0.2223	0.0773	0.1288	**0.0648**	0.2138
heart	0.7630	**0.7536**	0.7575	0.9994	0.3761	0.3749	**0.3646**	0.7642

smaller distance than X_l^*. Compared to the case of subspace clustering, candidates' relative positions to the true clustering in the full space are more complicated. Thus it is more difficult to predict the performance of the combined clustering by the two methods. However, on all datasets, X_l^* is always closer to T than X_l^+, which suggests X_l^* is less sensitive to the variation of candidates than the combined clustering.

4.9 Summary

In this chapter we addressed two basic problems in consensus clustering. First we proposed a series of entropy-based distance measures for comparing clusterings. It only involves set intersection operation and is independent of the data type and structure in question, since the input to our problem is a set of cluster labelings, rather than the original data themselves. We showed that they satisfy some of basic properties a legal distance function requires. Given a set of candidate clusterings, they enable us to find the local centroid candidate defined as the one with the smallest average distance to them. We also discussed search methods for the global centroid clustering. Under certain conditions, the centroid clustering will probably be closer to the true partition than other candidates. This assertion was demonstrated on both artificial and real datasets, with candidate clustering either from full space or subspace.

It is important to note that the key factor in the success of our combining methods is the relative positions of candidate clusterings w.r.t. the true clustering. Analogous to the requirement of the powerful but diverse classifiers in multiple classifier system, we hope that all candidate clusterings are not too bad and center evenly around the true clustering. When this constraint is dropped, there is no guarantee that the combined clustering will get closer to the true clustering, though probably the local centroid candidate would still be at least middle-ranked.

Chapter 5

FINDING PATTERN-BASED OUTLIERS

5.1 Introduction

In this chapter, we turn our attention away from finding clusters for the majority of the data to outlier detection that targets those exceptional data whose pattern is rare and different from the general pattern shown by the majority of the data. We illustrate that besides high density clustering, there is another pattern, low density regularity. Thus, there are two kinds of corresponding outliers w.r.t. them. Then we propose two techniques, one used to identify the two patterns, the other used to detect outliers w.r.t. them.

The chapter is organized as follows. In the rest of this section we give motivation and problem formulation. Related work is reviewed in Section 5.2. In Section 5.3 , first we show two patterns, high density clustering and low density regularity. Then, under assumption of uniform distribution inside clusters, we propose a technique to identify these two patterns based on the volume of the sphere. Also based on this random variable, we develop in Section 5.4 an approach to detecting local outliers with its sample variance. After discussing some formal evaluation criteria in Section 5.5, we report experimental results in Section 5.6 on both synthetic and real datasets. Section 5.7 concludes this chapter with a summary.

5.1.1 Motivation

In contrast to traditional clustering aiming to find general pattern for the majority of data, outlier detection targets the finding of rare pattern for the minority of the data whose behavior is very exceptional compared to other data. Although the meaning of outlier seems straightforward to many people, there is no universally accepted formal definition and only some intuitive interpretations are available in the literature. A well-known definition of outlier was given by Hawkins [58] who defined it as an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism. A similar definition also appeared in Barnett and Lewis's book [7] which stated that an outlier is an observation that appears to be inconsistent with the remainder of that set of data. Beckman and Cook [9] also gave an alternative definition of outlier as a contaminant or a discordant observation, where a discordant observation refers to any observation that appears surprising or discrepant to the investigator, and a contaminant is any observation that is not a realization from the target distribution.

Using the above general definitions, we always imply some pattern w.r.t. which we declare some data points are outliers. This pattern is followed by the global/local majority of the data and is breached by the outliers. In detail, it is embodied by 'other observations 'in Hawkins's definition, by 'the remainder of that set of data' in Barnett and Lewis's definition, and is the synonym of 'the target distribution' in Beckman and Cook's definition.

Although outliers are often treated as noise or error in many operations, such as clustering, and discarded, they may have potential causes and bear useful information that cannot be mined from other data that reside deeply inside clusters. It is not unusual that one man's noise is another one's signal. After identifying possible outliers,

we may go further to study the underlying reasons why they happen and this knowledge may be profitable. For instance, outliers may be produced by an incorrect assumption of distribution. In such situations, further investigation for outliers can lead to a more appropriate statistical model, which, in turn, leads to a more appropriate statistical inference. Occasionally, the presence of outliers indicates more information than being assumed. This is often true in exploratory data analysis, for at least three structures, cluster, complete spatial random (Poisson) process and regular spacing are often simultaneously present in the data. So in a way, finding outliers is at least as important as finding general patterns like clustering structure. Outlier detection has already found practical application including discovering crime in e-commerce, discovering computer intrusion, detecting credit card fraud, etc.

There are many similar problems in other fields. For instance, in association rule mining, an outlier is an interesting rule and the outlier factor is the interestingness. The rule's interestingness can be measured in terms of its unexpectedness, i.e., how much it changes the current belief of the whole system of all mined rules so far [92, 119]. In pattern classification, data from rare classes can be regarded as outliers [1]. The outlier factor is associated with the increase in the error rate after introducing it, i.e., how much it defies the current constructed classifier based on those data from major classes.

5.1.2 Problem Formulation

Now we give the formal formulation of outlier detecting problem. Given a labeled dataset partitioned as outliers and non-outliers, the problem of detecting outliers is essentially an unsupervised 2-class classification problem with class labels unknown to the classifier.

- Given

  A dataset $X = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{n}$, where $\mathbf{x}_i \in \Re^d$ and $\omega_i \in \{\omega_o(\text{outlier}), \omega_n(\text{non-outlier})\}$.

- Find

  A mapping function $f : X \rightarrow \Re^+$, i.e., f maps each data point to a positive value regarded as the outlier factor, the degree of outlyingness.

- Objective

  $\forall \mathbf{x}_i, \mathbf{x}_j, \omega_i = \omega_o \bigwedge \omega_j = \omega_n \Rightarrow f(\mathbf{x}_i) > f(\mathbf{x}_j)$, i.e., any outlier's factor must be greater than all non-outliers' factors.

- Constraint

  Class labels $\{\omega_i\}_{i=1}^n$ are unknown to the learner.

If the ranking of outliers is available, a more demanding requirement would be that if an outlier is known to be more outlying than another outlier, its outlier factor by f must be greater than another's.

5.2 Related Work

Most outlier detection techniques handle outliers where all attributes of the object are treated equally, i.e., each object with d continuous attributes is regarded as a point in $\Re^d$. In the rest of this chapter, we will sometimes use word like object, data point and event interchangeably, provided no ambiguity occurs. Generally speaking, outlier detection techniques can be divided into the following categories: distribution-based, depth-based, distance-based and density-based.

Distribution-based methods often handle one dimensional data and are mainly developed in the statistical field [7]. They assume a statistical distribution such as Gaussian and try to fit the data to the model by estimating the parameters such as mean and variance from the data. They vary in terms of type of distribution, number of outliers to be identified and type of outliers. Then they employ a test based on the distrib-

ution property to identify outliers w.r.t. this distribution. For a dataset of n values, $\{x_i : i = 1...n\}$, let $\hat{\mu}$ denote sample mean and $\hat{\sigma}$ sample standard deviation, then the z-score of a point x is $z(x) \equiv (x - \hat{\mu})/\hat{\sigma}$. For data from Gaussian like distributions, $z(x) \sim N(0, 1)$, and one popular test labels x outlier if its absolute z-score exceeds 3, i.e., $|z(x)| > 3$. Obviously, this test targets those points on the distribution tail. [130] gave an online approach, using Gaussian mixture to model the data. As new datum is read, the model is modified to maximize likelihood and the new datum's outlyingness is measured in terms of difference between the new and the original distribution. In reality, prior knowledge about the distribution of the dataset is not always available. Furthermore, it is hard to justify model selection in advance, e.g., Gaussian over exponential.

Depth-based approaches [113, 75] employ computational geometry to compute different layers of convex hulls and declare those objects in the outer layer as outliers. However, they suffer from the dimensionality curse and cannot cope with large dimension [24].

The remaining two categories are capable of dealing with multi-dimensional data and are mainly developed in the database community recently. They are closely related to the corresponding clustering algorithms that try to find the general pattern followed by the majority of the data. In fact, given a clustering algorithm with a function to measure its clustering quality, a naive algorithm for calculating outlier factor can assign each point a value that equals the absolute difference between the original clustering quality and the new clustering quality after removing that point. Further consideration will also include the clustering complexity, e.g., number of clusters. This is related to finding the best model fitted to the data with the criterion of minimum description length, where clustering quality corresponds to the likelihood of the data and clustering

complexity corresponds to the model complexity.

Distance-based techniques distinguish points which are likely to be outliers from others based on the number of points in their neighborhood. They do not assume any prior distribution of the data and limit the counting of points to the neighborhood of each point. These properties make them suitable for finding outliers in large datasets. Corresponding to clustering algorithms that find convex clusters [82, 100], one well-known technique is DB(p, d)-outlier [85], where a point in a dataset T is an outlier if at least p fraction of points in T lie greater than distance d from it. A special case of DB(p, d)-outlier is proposed in [106], where the distance to the k-th nearest neighbor is used to rank the outlyingness. The strength of this definition includes simplicity and capture of the basic meaning of Hawkins' definition. However, it cannot handle data with different local densities and hence can only find global outliers. Besides, the user's parameters, such as p, d, k, are hard to determine beforehand.

Density-based approaches focus on the local density comparison only with the immediate neighbors. They come in two classes, subspace and full space. Sometimes, a point could reside in a low density region only in a subspace, which is obtained by projecting the original full space onto one of its subsets. Corresponding to clustering algorithms capable of finding clusters in subspace [2], [1] considered such situations and searched for all possible subspaces where there are regions with much lower density than the rest of the subspace. All points in those low density regions are declared as outliers. [84] also considered subspace and tried to explain why a point is outlying in terms of intensional knowledge by finding the minimal subspace where it is outlying for the first time.

5.2.1 Local Outlier Factor

Because we mainly compare our approach against local outlier factor (LOF) [15], we introduce it here in some detail. Corresponding to clustering algorithms capable of

finding arbitrary shape clusters [25, 5] in the full space, Breunig et al. [15] proposed the notion of LOF, which measures the degree of outlyingness, based on the difference in the local density of a point and its k nearest neighbors. Generally speaking, DB(p, d)-outlier can only find global outliers that lie far away from all spherical clusters. As demonstrated in [15], DB(p, d)-outlier cannot detect local outliers w.r.t. a neighboring dense cluster in presence of another very sparse cluster. The reason is that although the local density of the outlier can be lower than those inside the neighboring high density cluster, it may be comparable to those inside the sparse (low density) cluster. However parameters are tuned in DB(p, d)-outlier, to successfully predict the true outlier, a large portion of points in the sparse cluster will also be classified as outliers. LOF solves this problem by thinking locally, i.e., comparing local density of the outlier only with those of its neighboring points. Essentially LOF consists of three definitions in Eqs. (5.1,5.2,5.3). Eq. (5.1) defines the reachability distance of an object p w.r.t. another object o, denoted by $rd_k(p, o)$, where $d_k(o)$ denotes the distance from o to its k-th nearest neighbor and $d(p, o)$ denotes the distance from p to o. Local reachability density of p, denoted by $lrd_k(p)$, is defined in Eq. (5.2), where $N_k(p)$ denotes k-th order neighborhood of p. For those p close to o, i.e., $d_k(o) > d(p, o)$, the usage of reachability distance instead of pure distance smooths $lrd_k(p)$ by making o's contribution the same, i.e., always using $d_k(o)$ instead of $d(p, o)$. LOF of p w.r.t.k is defined in Eq. (5.3) as the average ratio of its neighbor's density over p's density. If points inside the cluster are approximately uniformly distributed, their local reachability density will be similar and hence their LOF will be close to 1. For an outlier outside the cluster, its local reachability density will be lower than those of its neighbors inside the cluster and its LOF will be higher than 1. So LOF ranks points in descending order of their LOF and those on the top are declared as local outliers.

$$rd_k(p,o) \equiv \max\{d_k(o), d(p,o)\} \tag{5.1}$$

$$lrd_k(p) \equiv \frac{|N_k(p)|}{\sum_{o\in N_k(p)} rd_k(p,o)} \tag{5.2}$$

$$LOF_k(p) \equiv \frac{\sum_{o\in N_k(p)} lrd_k(o)/lrd_k(p)}{|N_k(p)|} \tag{5.3}$$

The weakness of LOF is that it cannot detect outliers whose local density is higher, not lower, than those inside the neighboring pattern. Such a pattern may consist of a set of regularly spaced points that have lower densities than their neighboring outliers. The introduction of the outlier significantly breaks the regularity and increases the local densities.

5.3 Patterns Based on Complete Spatial Randomness

5.3.1 Complete Spatial Randomness

As we mentioned above, whenever we declare a data point an outlier, we always imply some pattern w.r.t. which it is outlying. According to Webster's dictionary, a pattern is 'a natural or chance configuration, or a reliable sample of traits, acts, tendencies, or other observable characteristics'. Extremely speaking, anything can be a pattern, or show some kind of pattern, to be more exact. For example, a point $\mathbf{x} \in \Re^d$ can define a pattern with itself, i.e., any point $\mathbf{y} \in \Re^d$ follows this pattern if $\mathbf{y} = \mathbf{x}$ and otherwise it is an outlier. Complete Spatial Randomness (csr) refers to a lack of structure in the spatial point process, where events (points regarded as realization of events) are uniformly distributed in the study region $A \subset \Re^d$. For any sub-region $B \subset A$, the probability that it contains at least one event is equal to the ratio of its volume over the total volume, i.e., $|B|/|A|$, where $|\cdot|$ denotes volume. This probability is independent from B's location and shape. This kind of spatial point process is also called homogeneous Poisson process, because the number of events in B follows a Poisson distribution and

the intensity is the same everywhere in the study region. Formally, if $d\mathbf{s}$ denotes an infinitesimal region located at $\mathbf{s}$, $N(B)$ denotes the number of events in $B \subset \Re^d$, then a point process is a homogeneous Poisson process provided the following conditions hold [20]:

1. $$\lim_{|d\mathbf{s}|\to 0} \frac{1 - P(N(d\mathbf{s}) = 0)}{|d\mathbf{s}|} = \lim_{|d\mathbf{s}|\to 0} \frac{P(N(d\mathbf{s}) = 1)}{|d\mathbf{s}|} = \lambda$$

   where λ is the intensity (density) of the process and it is the same for all $\mathbf{s} \in \Re^d$. If it is replaced by $\lambda(\mathbf{s})$, i.e., a function of $\mathbf{s}$ and could vary, then the process becomes an inhomogeneous Poisson process.

2. $N(d\mathbf{s}_1), ..., N(d\mathbf{s}_m)$ are statistically independent for any disjoint sequence of regions $d\mathbf{s}_1, ..., d\mathbf{s}_m$.

$N(B)$ can be approximated with a binomial distribution with parameters $n = |B|/|d\mathbf{s}|, p = \lambda|d\mathbf{s}|$. As $|d\mathbf{s}| \to 0$, $N(B)$ converges to a Poisson distribution with mean $\lim_{|d\mathbf{s}|\to 0} np = \lambda|B|$. Apparently it does not depend on B's location and shape. Furthermore, given $N(B) = n$, n evens are independently and identically uniform distributed over B. For any two disjoint regions B_1 and B_2, $N(B_1)$ and $N(B_2)$ are independently Poisson distributed. A particular realization of a homogeneous Poisson process with $N(B) = 100$ and $|B| = 10 \times 10$ is given in Fig. 5.1(a). Note that it may seem clustered to untrained eyes due to its inherent randomness. If we strictly place those points with equal interval, it forms another structure called regular spacing.

5.3.2 Clustering and Regularity

A cluster with arbitrary shape can be defined as a set of points with similar densities that are significantly higher than those of points in its immediate surrounding area. Both

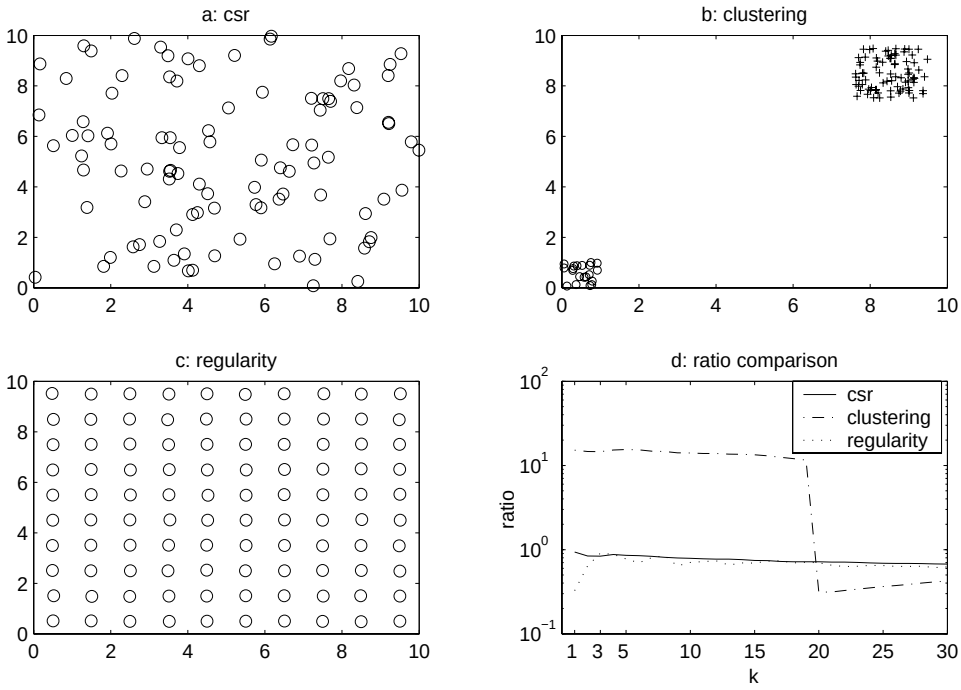

Figure 5.1: (a-c) illustrate three structures respectively, complete spatial randomness, clustering and regularity. (d) shows their ratios vs k.

homogeneous and inhomogeneous Poisson processes have been used for cluster analysis in classification of remote sensing images [107]. Such a cluster has two properties [25]: maximum and connectivity. It is maximal in that any extension to it by including neighboring additional points will lead to a significant decrease in overall density. It is connective in that for any two points belonging to the cluster, there is a path linking them which consists only of the cluster points. Two clusters C_1 and C_2 are shown in the lower left and upper right corners of Fig. 5.1(b), where C_1 has 20 points uniformly distributed in a 1×1 area and C_2 has 80 points uniformly distributed in a 2×2 area. Compared to csr, clustering means that points tend to attract one another and consequently, the average nearest neighbor distance is smaller than that of csr.

Fig. 5.1(c) illustrates 100 points regularly spaced with approximately 1 intervals in both horizontal and vertical directions. Note that we add Gaussian noise with zero mean and small deviation ($\sigma = 0.01$), after positioning points at constant 1 intervals.

In spite of the Gaussian noise, the difference between it and csr in Fig. 5.1(a) is still obvious. In a way, regular spacing can be regarded as a special cluster in that the points are distributed so uniformly that it shows too little randomness. Compared to csr, regularity means that points tend to push one another. As a result, the nearest neighbor distance is approximately the same for all points and is larger than its counterpart in csr. Besides, for each point and some small j (e.g., $j = 4$ in Fig. 5.1(c)), its k-th ($k \leq j$) nearest neighbor distances are usually also the same.

In addition to csr, clustering and regularity, with inhomogeneous Poisson process at hand, any pattern (distribution) can be described, as long as we can divide study area into enough sub-areas each of which can be modeled by csr. For instance, points from Gaussian distribution can be partitioned into subsets each of which is a contour in terms of pdf. Over each contour, points follow csr.

5.3.3 Identifying Clustering and Regularity

Let V_k denote the random variable of the hyper-sphere volume centered at a randomly chosen point in $B \subset \Re^d$, with radius equal to the distance to its k-th nearest neighboring object. Note that it does not matter whether there is an event (object) happening at that point location. Imagine we inflate a sphere centered at that point by increasing the radius, as more and more nearby objects get enclosed, V_k is the volume of the sphere reaching the k-th object. By assuming the distribution of the objects follows csr (homogeneous Poisson process) with constant intensity λ, the random variable V_k actually has a gamma distribution with parameter (k, λ), i.e., $V_k \sim \Gamma(k, \lambda)$ [111]. If the randomly chosen point above is replaced by a randomly chosen object, the distribution of the corresponding random variable remains the same, specified by its pdf in Eq. (5.4) ($\Gamma(\cdot)$ is the Gamma function), together with expectation and variance given by Eq. (5.5).

$$f(v_k) \equiv \frac{\lambda e^{-\lambda v_k}(\lambda v_k)^{k-1}}{\Gamma(k)} \tag{5.4}$$

$$E(V_k) = \frac{k}{\lambda},\ \mathrm{Var}(V_k) = \frac{k}{\lambda^2} \tag{5.5}$$

Based on the expectation, we propose a technique to identify the data structure by telling us whether it is csr, clustering or regularity. Furthermore, in the case of clustering with csr inside each cluster, it can tell the minimum cluster size. Given a dataset $\{\mathbf{x}_i \in \Re^d\}_{i=1}^n$, after collecting the volume $V_k = \pi^{d/2} R_k^d / \Gamma(1 + d/2)$ (R_k is the distance to the k-th nearest neighbor) for each datum and estimating the total intensity λ, we compute the ratio of the expectation of V_k over the observed one (averaging V_k for all data) and compare this ratio to 1, as in Eq. (5.6).

$$R(k) \equiv \frac{k/\lambda}{\frac{1}{n}\sum_{i=1}^{n} V_k} \tag{5.6}$$

$$R(k) \approx \frac{k/\lambda}{\frac{1}{n}\sum_{j=1}^{m} \frac{n_j k}{\lambda_j}} \tag{5.7}$$

The ratio R is obtained at multiple k. Then we can draw a figure of R versus k and identify the structure based on the following three properties:

1. If R is approximately close to 1 at all k, the data structure is csr.

2. If R is significantly less than 1 at small k, e.g., $k = 1, 2$, the pattern is regularity. Because the nearest neighbor distance of regularity is larger than csr, such relation also holds for the volume.

3. If R is significantly greater than 1 at many k, especially at small ones, the pattern is clustering. The reason is that its nearest neighbor distances are smaller than csr, which also leads to smaller volume at small k. Besides, if there are multiple

clusters, R will initially remain nearly constant as k grows, and drop sharply when k reaches the minimum cluster size.

The ratio R for three datasets in Fig. 5.1(a,b,c) is illustrated in Fig. 5.1(d) with $\hat{\lambda} = 100/(10 \times 10)$. As expected, R for csr in Fig. 5.1(a) is close to 1 for all k. For regularity in Fig. 5.1(c), R is significantly smaller than 1 at $k = 1, 2$ and close to 1 at $k = 3$. It means that under csr, the average distance to the 3rd nearest neighbor is close to 1. For clustering in Fig. 5.1(b), R's curve is relatively flat as $k < 20$, and drops radically at $k = 20$, the smaller cluster C_1's size. The reason is that at $k = 20$, the 20-th nearest neighbor of every point in C_1 is in C_2, which means their V_k no longer follows $\Gamma(k = 20, \lambda = 20/(1 \times 1))$. Generally, suppose the dataset consists of m disjointed clusters $\{C_j(n_j, \lambda_j)\}_{j=1}^{m}$, where n_j and λ_j are the j-th cluster size and intensity, and $n_1 \leq ... \leq n_m$. Under csr inside every cluster, we can approximate the sample mean of V_k, the denominator in Eq. (5.6), with the denominator in Eq. (5.7). That is, replacing the sum of V_k in every cluster with the expected value. Consequently, R in Eq. (5.7) is independent of k and remains constant till the replacement is no longer valid at $k = n_1$ when the k-th nearest neighbor of every point in C_1 is no longer in C_1 and the corresponding V_k no longer follows $\Gamma(k, \lambda_1)$.

5.4 Detecting Pattern-Based Outliers

A data point could be outlying w.r.t. a nearby high density pattern cluster because its own density is relatively low. This case is shown in Fig. 5.2(a), where there are two clusters, one dense C_1 and a sparse one C_2. Densities illustrated in Fig. 5.2(b) are obtained with a Gaussian kernel function $f(x) = \sum_{i=1}^{n} \exp(-d^2(x, x_i)/(2\sigma^2))$, where $\sigma = 1$ and $d(x, y)$ denotes the Euclidean distance between x and y. Point O_2 is a global outlier because its density is lower than both clusters and it can be detected by both

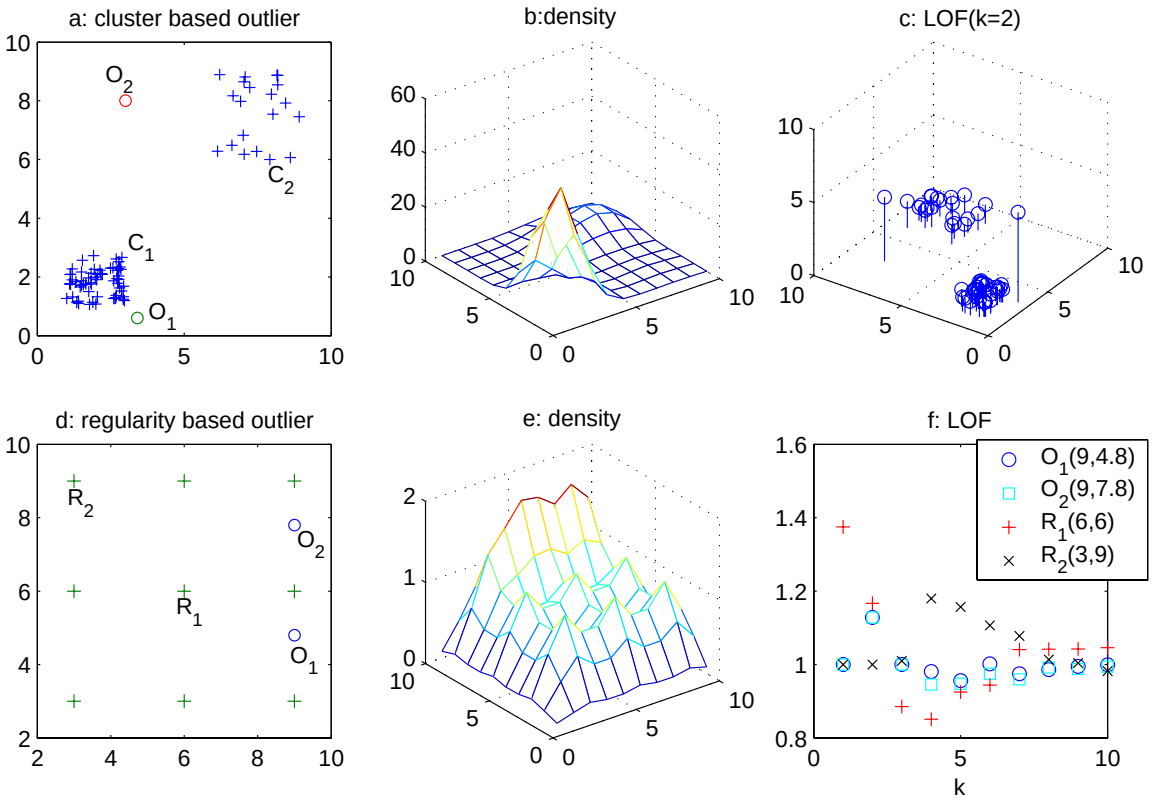

Figure 5.2: (a-c) illustrate cluster-based outliers, their density, and LOF ($k = 2$). (d-f) show regularity-based outliers, their density, and LOF ($k = 1, ..., 10$).

DB-outlier and LOF. Point O_1 is a local outlier w.r.t. C_1 because its density is lower than C_1 but comparable to C_2. Only LOF can detect it, as shown in Fig. 5.2(c). On the other hand, a data point could also be outlying w.r.t. a nearby low density pattern regularity, because its own density is relatively higher than neighboring points belonging to the regularity. This situation is shown in Fig. 5.2(d) where two outliers, O_1 and O_2, have densities higher than most of points of the pattern, a 3×3 grid of nine points, as demonstrated in Fig. 5.2(e) with the same kernel function. Fig. 5.2(f) proves that LOF cannot detect them by making their outlier factors simultaneously higher than all regularity points. In fact, R_2's LOF is consistently higher than that of both at all k except $k = 2$ where R_1 takes the lead.

Combining the two situations, we can conclude that a data point may be outlying

because its density is lower (higher) than a nearby high (low) density pattern. In other words, it is outlying because its density is different from those of most of neighbors belonging to the pattern. At this time the sample variance of R_k and consequently V_k is expected to be high. This observation leads to our approach to detecting local outliers based on Variance Of Volume (VOV). First, we formally define the k-th nearest neighbor distance $d_k(x)$ and the k-th order neighborhood $N_k(x)$ in case of multiple data at the same distance to the current query data point. For a dataset $X = \{x_i\}_{i=1}^n$, $d_k(x_i)$ is the distance $d(x_i, x)$ from x_i to another data point $x \in X$ with the following two conditions:

1. $|\{x : x \in X - \{x_i\}, d(x_i, x) \leq d_k(x_i)\}| \geq k$,

2. $|\{x : x \in X - \{x_i\}, d(x_i, x) < d_k(x_i)\}| < k$.

Consequently, $N_k(x_i) \equiv \{x : x \in X, x \neq x_i, d(x_i, x) \leq d_k(x_i)\}$. Then, our local outlier factor VOV can be computed as follows.

1. For each data point $x_i, i = 1...n$, retrieve its k-th neighborhood $N_k(x_i)$. For each data point $x \in x_i \cup N_k(x_i)$, compute V_k, the hyper-sphere volume centered at it with radius equal to $d_k(x)$, the distance to the k-th nearest neighbors.

2. Compute the sample variance of V_k and assign it as the VOV outlier factor to x_i.

The resulting formal definition of VOV is given by Eq. (5.8), with $N_k^+(x_i) \equiv x_i \cup N_k(x_i)$.

$$\begin{aligned} \overline{V_k}(x_i) &\equiv \frac{\sum_{x \in N_k^+(x_i)} V_k(x)}{|N_k^+(x_i)|} \\ VOV(x_i) &\equiv S^2(x_i) = \frac{\sum_{x \in N_k^+(x_i)} (V_k(x) - \overline{V_k}(x))^2}{|N_k^+(x_i)| - 1} \end{aligned} \tag{5.8}$$

5.4.1 Properties of VOV

The sample variance S^2 is itself a random variable. For data belonging to the pattern, it is preferred that $E(S^2)$ be smaller than those of outliers. Besides, $\mathrm{Var}(S^2)$ is also preferred small, which is achieved by usage of reachability distance instead of pure distance in LOF. If the pattern is regularity, it is easy to see that for some appropriately chosen small k, VOV is 0 for pattern points (approximately 0 if data are approximately regularly spaced). If the pattern is clustering, for simplicity, we assume $|N_k^+(x_i)| = k+1$, since for high dimensional data in reality, it is rare that multiple data stand at the same distance from another data point. In this case, for cluster (csr inside with intensity λ) points, $E(S^2) = k/\lambda^2$. If k is relatively large, gamma distribution can be approximated by Gaussian distribution and it can be shown that $\lambda^2 S^2$ follows a chi-squared distribution χ_k^2 with k degrees of freedom [111], so $\mathrm{Var}(S^2) \approx 2k/\lambda^4$.

From S^2's expectation and variance, we can see that k cannot be large. On the other hand, k cannot be too small. Suppose there are two outliers closest to each other, then their VOV are both 0 at $k = 1$. A method to choose k is based on the figure of ratio vs k in Eq. (5.6), where we use it to identify patterns. Based on that figure, we can find the minimum cluster size and therefore, k can be chosen at a value a little less than the minimum cluster size but still larger than the outlier cluster size, if multiple outliers really lie together. At that value k, for cluster points, their k-th nearest neighbors are still in the same cluster and hence V_k still follows a gamma distribution. For outliers, their k-th nearest neighbors are expected to lie in the nearby clusters and V_k does not follow a gamma distribution, Otherwise, those outliers themselves form a cluster of size $k + 1$ and it is not reasonable to regard them as outliers.

The remaining problem is how to estimate the total intensity or equivalently, how to estimate the volume of bounding region that encloses the dataset. Ideally, we should

compute the convex hull whose computation is complex and costs $O(n\log n)$, where n is the data size[24]. Because what we care is not the precise value of the ratio and the intensity ($\hat{\lambda}$ is fixed in Eq. (5.6)), but how the ratio changes, i.e., the minimum cluster size k at which the ratio drops sharply. We can approximate it by selecting the minimum of the volume of the isothetic rectangle and the encompassing sphere [83], both of which can be computed in $O(n)$. The former is just a hyper-rectangle orthogonal to the axes, with the j-th side length being the difference between the maximum and the minimum of j-th attribute over n data. The latter is the hyper-sphere centered at the midpoint of the main diagonal of the rectangle with radius equal to the half diagonal length.

As for time complexity, VOV is similar to LOF and takes $O(n \times (k\text{NN} + k))$ time, where kNN denotes the time for a k nearest neighbors query. The dominant part, $O(n \times k\text{NN})$, is spent in collecting V_k and it depends on the particular implementation of k nearest neighbors query, e.g., [48, 112, 10]. The remaining part $O(nk)$ is used for computing the sample variance of V_k.

5.5 Evaluation Criteria

The criteria evaluating outlier detection approaches can be divided into two parts: efficiency and effectiveness. Good efficiency means the technique should be applicable not only to small databases of just a few thousand objects, but also to even larger databases with millions of objects. Time complexity of VOV is similar to LOF and its computation can be divided into two steps. In the first step, VOV needs to retrieve the k-th order neighborhood $N_k(x)$ for each data point x together with their k-th nearest neighbor distances. The running time of this step mainly depends on the time for a k nearest neighbors query. In the second step, VOV computes the sample variance of the corresponding volume derived from the distance (radius) for each data point's augmented

k-th order neighborhood $N_k^+(x)$ (including itself) and it takes time $O(n)$.

As for effectiveness, considering that the final user is human, a good approach should require as few input parameters from the user as possible. Besides, these parameters should have intuitive meaning (such as k) and thus make it easy for the user to determine. Ideally, a good approach will automatically detect the various patterns and the corresponding outliers.

At this time, we should discuss some formal criteria. Given a labeled dataset partitioned as $D = D_O \cup D_N, D_O \cap D_N = \emptyset$ where D_O and D_N denote outliers and non-outliers respectively, for any outlier detection method $M(\theta)$ where θ denotes its parameter vector to be determined, we say $M(\theta)$ is consistent with D if we can find some particular estimate (values)$\hat{\theta}$ for θ such that $M(\hat{\theta})$ can correctly partition D [96]. Apparently, we prefer a method M that is consistent with more labeled datasets, as long as that labeling is reasonable. For method comparison, $M_1(\theta_1)$ is said to be more general than $M_2(\theta_2)$ w.r.t. an unlabeled dataset D if for any partition of D with which $M_2(\theta_2)$ is consistent, $M_1(\theta_1)$ is also consistent with that partition. Naturally, we favor a more general method. Similarly, many concepts in computational learning theory can also be applied here. For instance, we say $M(\theta)$ shatters an unlabeled dataset D if for any 2-class partition of D, we can always find some $\hat{\theta}$ such that $M(\hat{\theta})$ is consistent with that partition of D. Thus we can define M's VC-dimension as the maximum size of D that can be shattered by M. VC-dimension describes the complexity or flexibility of M. However, high VC-dimension is not always preferred, for among the $2^{|D|}$ partitions of D, many are illogical, e.g., $|D_O| = |D| - 1, |D_N| = 1$. So a practical requirement for M may be that it can detect a finite number of fixed patterns and allow the user to specify in advance the patterns on which he/she hopes the detected outliers will be based.

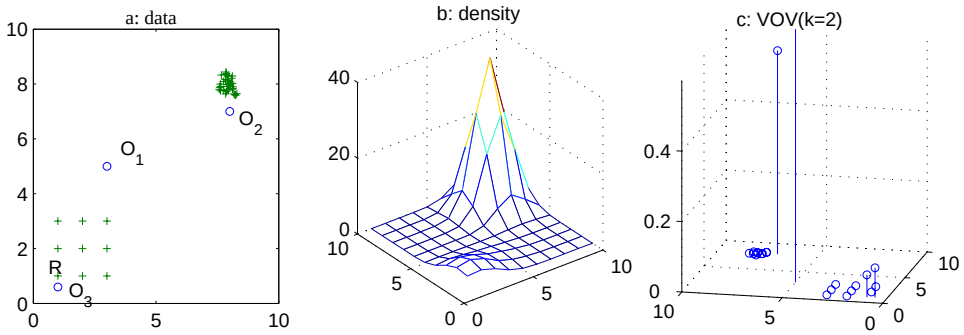

Figure 5.3: (a) shows a dataset with both cluster and regularity-based outliers. Its density and VOV ($k = 2$) are illustrated in (b,c) respectively.

5.6 Experimental Evaluation

We test our VOV on both synthetic data and real data. On the former, we show that with appropriately chosen k, VOV can simultaneously detect local outliers w.r.t. high density cluster and low density regularity. On the latter, we compare VOV against LOF on three datasets from the UCI repository.

5.6.1 Synthetic Data

A dataset is illustrated in Fig. 5.3(a) with a cluster in the top right corner and a regularity in the bottom left corner. In addition, there are three outliers, including a global outlier O_1, a local cluster-based outlier O_2 and a local regularity-based outlier O_3. The density with Gaussian kernel is shown in Fig. 5.3(b) and their VOV outlier factors are shown in Fig. 5.3(c) with $k = 2$. We can see VOV successfully separates outliers from pattern points and consequently is consistent with this labeled dataset. Pattern point $R(1, 1)$ has the largest VOV over pattern points and this is reasonable, because its density is greatly increased by the presence of the neighboring outlier O_3. Detailed VOV values of outliers and R are shown in Table 5.1.

Table 5.1: VOV of outliers O_i and R.

O_1	O_2	O_3	R
52.6379	0.5779	0.0842	0.0632

5.6.2 Real Data

We choose from the UCI repository three datasets, ionosphere, Wisconsin diagnostic breast cancer and Pima Indians diabetes, which vary a lot in data size and dimension. All of them are of binary class, and we select all data from the majority class as non-outliers and select the first m data in the original order from the minority class as outliers such that in the resulting dataset the ratio of non-outliers over outliers is $9:1$.

First, we draw the figure of ratio vs k in Figs. 5.4(a,d,g). Compared to the corresponding csr with the same bounding region, we can see this ratio is far lower than 1, i.e., the average k nearest neighbor distances are much larger than those under csr. This confirms the assertion of sparsity of high dimensional data in [1]. For ionosphere data, the maximum ratio is achieved at $k = 7$. For the other two, the ratio keeps decreasing. So we choose $k = 3, 7$ for subsequent comparison.

After choosing k, both VOV and LOF provide a ranking of data in decreasing outlier factor. We can choose top $100p\%$ data $T(p)$, compare them to the true outliers O (those 10%) by computing recall $|T(p) \cap O|/|O|$ and precision $|T(p) \cap O|/|T(p)|$. In this case, a larger recall also means a larger precision and we illustrate recall in Fig. 5.4. The larger the recall, the better. To compare VOV and LOF, we concentrate on two aspects. The first aspect is recall at small p, because it is the common practice in reality that we usually select some top predicted outliers for further investigation. Furthermore, the smaller p is, the more important the corresponding recall. The other aspect is the minimum of p at which VOV and LOF achieve full(100%) recall. From these two aspects,

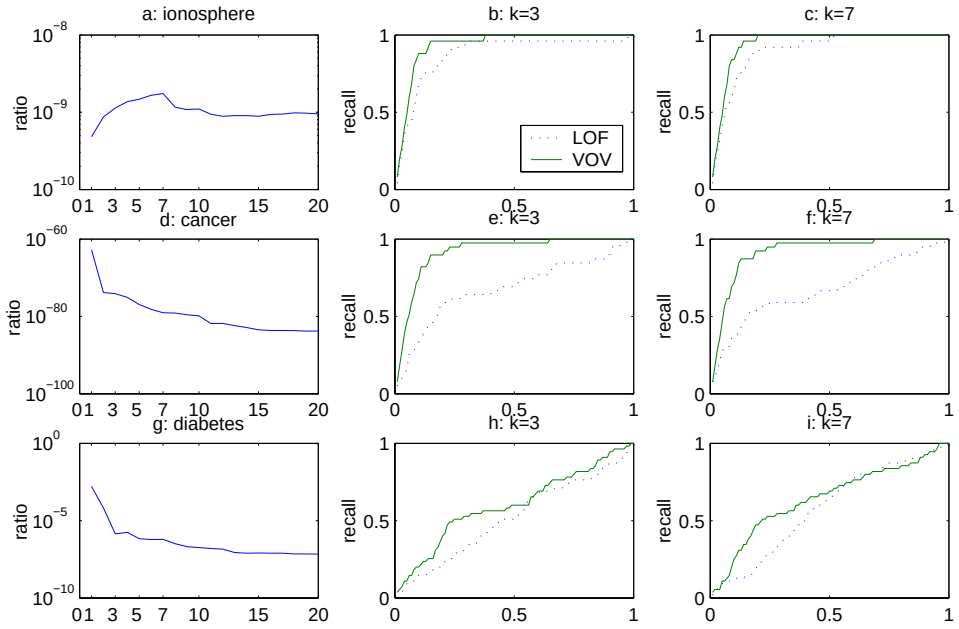

Figure 5.4: (a) shows the ratio for ionosphere. Its LOF vs VOV is plotted in (b) for $k = 3$ and (c) for $k = 7$. The corresponding values for cancer and diabetes are shown in the middle and bottom rows, respectively.

we can see that VOV is consistently and significantly better than LOF on ionosphere and cancer data, which implies these two datasets coincide with our definition of outliers and assumption of csr inside clusters. As for diabetes data, our assumption is probably no longer valid; however, VOV is still much better than LOF on the recall of small p. VOV is consistently better than LOF at $k = 3$ and is slightly overtaken by LOF at $p \in [0.6, 0.8]$ with $k = 7$. These key values are shown in Table 5.2, including recall at small p around 0.1 and the minimum p at which VOV(LOF) achieves 100% recall.

To further analyze the prediction set, we divide $T(p)$ into three subsets: intersection of true positive ($T(p) \cap O$) between LOF and VOV, difference of true positive, and false positive ($T(p) - O$). Roughly speaking, true positive intersection includes those cluster-based outliers both LOF and VOV are able to detect. True positive difference of VOV can be interpreted by those regularity-based outliers that LOF fails to detect. The fraction of these three subsets at four values of p is shown in Fig. 5.5. We can see

Table 5.2: VOV vs LOF on the three datasets.

	$k=3$				$k=7$			
ionosphere: p	0.10	0.20	0.38	0.98	0.10	0.15	0.20	0.52
LOF	0.68	0.84	0.96	1.00	0.64	0.80	0.88	1.00
VOV	0.88	0.96	1.00	1.00	0.84	0.96	1.00	1.00
cancer: p	0.10	0.20	0.65	0.98	0.10	0.15	0.20	0.52
LOF	0.33	0.56	0.77	1.00	0.38	0.54	0.82	1.00
VOV	0.74	0.90	1.00	1.00	0.72	0.92	1.00	1.00
diabetes: p	0.10	0.20	0.30	0.90	0.10	0.20	0.96	1.00
LOF	0.15	0.20	0.33	1.00	0.13	0.22	0.96	1.00
VOV	0.20	0.40	0.53	1.00	0.25	0.47	1.00	1.00

that at all p LOF fails to capture some true outliers discovered by VOV. As p increases to 0.15, however, almost all true outliers predicted by LOF are also found by VOV.

In addition, since in $\Re^d$ the sphere volume V_k is derived from the radius R_k as $V_k = \frac{\pi^{d/2}}{\Gamma(1+d/2)} R_k^d = CR_k^d$, we can obtain R_k's density, expectation and variance, e.g., $E(R_k) = (\Gamma(k+1/d)/(k-1)!)(\lambda C)^{-1/d}$. Similarly, we can utilize the sample variance of R_k to measure outlyingness. Experimental results on the three datasets show it is slightly worse than that with V_k but still significantly better than LOF.

5.7 Summary

In this chapter, we first illustrated that there are at least two patterns, high density cluster and low density regularity. Therefore, there are two kinds of corresponding outliers w.r.t. them. Under assumption of csr inside clusters, we proposed a technique to identify them, based on the volume of the sphere centered at each data point with radius equal to its k-th nearest neighbor distance. Also based on the sample variance of this random variable, we developed a VOV approach to detecting outliers. Experimental results show our approach can simultaneously detect outliers w.r.t. both patterns and is better than LOF in terms of recall on the three real datasets from the UCI repository.

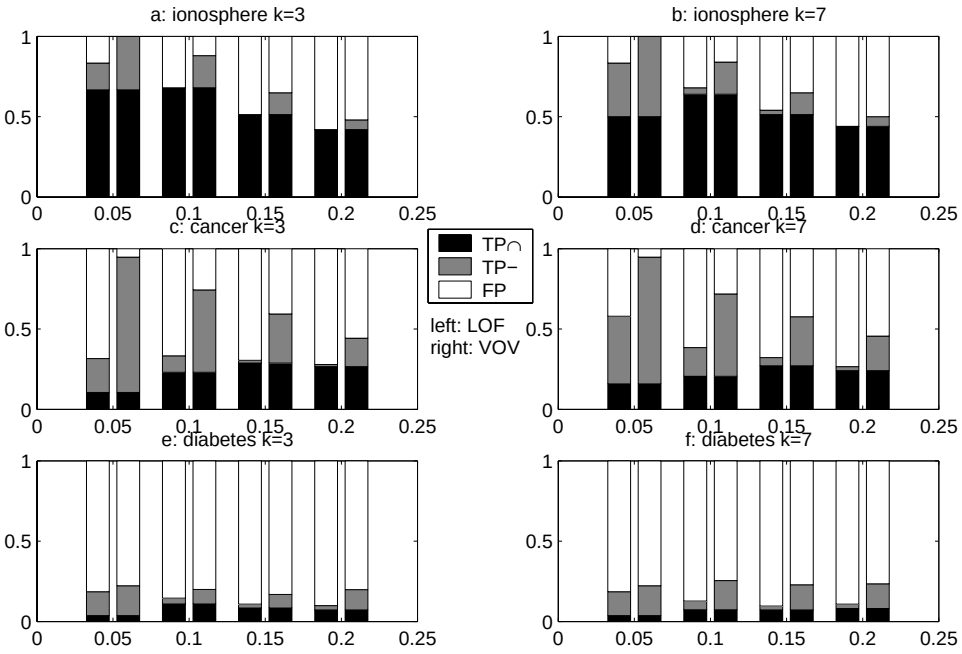

Figure 5.5: Comparison of makeup of prediction by LOF (left bar) and VOV (right bar). $TP\cap, TP-$ and FP denote intersection of true positive, difference in true positive and false positive, respectively.

One weakness of VOV is that its expectation for cluster points still depends on λ and it is expected that VOV for points inside very sparse (small λ) could be higher than a local outlier w.r.t. a dense cluster. A possible remedy to remove λ is to divide it with squared sample mean, i.e., $S^2/\overline{V_k}^2$, since $E(S^2) = k/\lambda^2$ and $E(\overline{V_k}) = k/\lambda$. However, this is only valid for cluster points and it is hard to interprete the sample mean in presence of outliers. Experiments show it leads to much poorer performance on the three real datasets. In a way, it confirms again the assertion of [1] that in the sparse high dimensional space, outliers from rare classes usually lie in regions of even lower densities.

Chapter 6

CONCLUSION AND FUTURE WORK

6.1 Major Results

This thesis has made several contributions to spatial data analysis, which are summarized below.

In Chapter 2, we proposed hidden fusion in radial basis function (RBF) networks for spatial regression. Assuming independent and identical distribution and ignoring spatial information, conventional RBF networks usually fail to give satisfactory results on spatial data. In contrast to input fusion, we pushed spatial autocorrelation further into RBF networks by fusing output from hidden and output layers. Empirical studies demonstrated the advantage of hidden fusion over others in terms of regression quality, MSE. Furthermore, compared to conventional RBF networks, hidden fusion does not entail much extra computation.

In Chapter 3, we developed a hybrid expectation-maximization (HEM) approach for spatial clustering using Gaussian mixture. The goal is to incorporate spatial information while avoiding much additional computation incurred by neighborhood EM (NEM) for E-step. In HEM, early training is performed via a selective hard EM till the penalized likelihood criterion no longer increases. Then training is turned to NEM, which runs only one iteration of E-step. Thus spatial information is incorporated throughout HEM,

which makes it achieve clustering results better than EM and comparable to NEM. Its complexity is retained between EM and NEM.

In Chapter 4, we continued to study clustering, but at a higher level. Consensus clustering aims to combine a given set of multiple partitions into a single consolidated partition that is closet to them. First we proposed a series of entropy-based functions for measuring distance among partitions. Then we developed two search methods for the global optimal partition based on similarity-based graph partitioning. Given a candidate set of partitions, the centroid partition will be probably top/middle-ranked in terms of distance to the true partition, which we demonstrated on a variety of datasets.

In Chapter 5, we turned our attention from the majority of the data to the rare outliers who cannot be assigned to any cluster. Most algorithms target those outliers with exceptionally low density, compared to nearby clusters of higher density. We showed that besides high density clustering, there is another pattern, low density regularity. Thus, there are at least two kinds of corresponding outliers w.r.t. them. We proposed two techniques, one used to identify the two patterns and the other used to simultaneously detect outliers w.r.t. them.

6.2 Future Work

6.2.1 Spatial Regression Using RBF Networks

Several issues are worth further study in spatial regression using RBF networks. One concerns the performance criterion. We employed MSE where all sites receive equal weight in the summation. In our RBF network model with data fusion, every site's prediction is actually a combination based on its own input and the prediction from its neighbors. Naturally we hope that the prediction is more accurate at those sites with more neighbors, since they contribute more often in others' prediction. So it may

be more appropriate to use a weighted least square criterion $(\mathbf{y} - \Phi\mathbf{w})^T A(\mathbf{y} - \Phi\mathbf{w})$, where A could be a diagonal matrix with i-th diagonal element proportional to site s_i's neighborhood size.

In HF2 ρ appears only once as the weighting coefficient for the virtual neighbor $W\mathbf{y}$ and results in lower MSE compared to conventional RBF networks. To improve performance further, we may try other types of hidden fusion, say, introducing a second weighting coefficient for $\Phi\mathbf{w}$, which leads to $\mathbf{y} = (1-\rho)\Phi\mathbf{w} + \rho W\mathbf{y}$.

Finally, there are other candidate places where spatial information can be pushed into RBF networks. For instance, the center selection, which is achieved with K-means in our work, plays a vital role in regression performance and different clustering techniques apparently would give different results [17]. However, they are all performed in the attribute space and no spatial information is taken into account. A reasonable anticipation is that data belong to the same center are also close in the spatial space, provided spatial continuity exits. A more ambitious requirement is that the center label can tell more about the dependent variable. This can be done by optimizing mutual information $I((Y,S),M)$ or conditional entropy $H(Y,S|M)$, where M denotes the unknown center label whose distribution needs to be estimated, Y denotes the dependent variable and S denotes the spatial location. To make computation feasible, Y needs to be discretized and S needs to be clustered, which poses additional challenges.

6.2.2 Spatial Clustering with HEM

There are several research directions of improving HEM for spatial clustering. First, as in most EM style algorithms, the final result of HEM depends on initialization. An online version of EM is introduced in [132] and its performance is invariant to initialization. However, it is impossible to directly apply that algorithm to our problem, for the penalty term cannot be factorized as likelihood. Second, it is worth trying other

penalty terms, such as the derivative of likelihood. The general requirement is that it should embody spatial information without entailing much trouble in optimizing the penalized new criterion. Finally, as in NEM, choosing penalty term coefficient β remains a main difficulty and it is highly desirable if we can automatically determine its optimal value. This value may be chosen independently for each site by automatically weighting its relative importance.

6.2.3 Online Approaches

The algorithms proposed in Chapters 2 and 3 to train mixture models for spatial regression and clustering are batch-based, that is, we need to feed all the data into the models simultaneously before the training can take place. For some real world applications, such complete and detailed information may be difficult and expensive, if not impossible to obtain. Take the election data for example. It is nearly impossible to take a nation-wide census within a very short time in a large country, so the results usually come sequentially and we may need to train the model with only partial data. Another scenario is that with a constructed model for a previous year, we need to train the model for a new year whose data are not quite different from the old ones, because econometric data generally vary slowly. In both, with only batch-based algorithms at hand, we can only discard the old model and train all over again with the new data, which is obviously uneconomical.

Such situations make it necessary to develop online approaches that are capable of dealing with sequential or real-time data. The general idea is to deemphasize the past data as new data come and, instead of discarding the old model, refine the model based on learned experiences and the new data. Such learning is closely related to the recurrent network with feedback loops [127] and reinforcement learning [77] with indirect and delayed rewards. Both are mainly developed for temporal learning, that is,

learning a sequence of data that is one dimensional in time. As Markov random fields generalizes the one dimensional Markov model/chain, to learn spatial data sequentially, we need to extend those algorithms from recurrent networks and reinforcement learning.

6.2.4 Consensus Clustering

There are two research directions for consensus clustering. One concerns the clustering distance function. All distances we developed in Chapter 4 are based on $d(X,Y) = H(X|Y)+H(Y|X)$, which is a special case of $d(X,Y) = \alpha H(X|Y)+(1-\alpha)H(Y|X)$ when $\alpha = 0.5$. If we know that some candidates are better than others, in computing distance to this set, it may be more appropriate to use different values for α to emphasize those better ones. We can also use different values to weigh $d(X,X^m)$ in $D(X,\{X^m\}_{m=1}^M)$.

The other direction is about search methods, which are discussed in some detail below.

At the resolution level of joint-clusters, some strategies for hierarchical clustering are readily available, e.g., agglomerative (bottom-up) and divisive (top-down). The agglomerative clustering starts at the bottom of clusters in $\left(H^1,...,H^M\right)$ and at each level recursively merges two selected clusters that leads to decrease in distance. Intuitively, we should first try those pairs that have non-empty intersection in some dimension, e.g., cluster (h_1^1,h_1^2) and cluster (h_1^1,h_2^2) intersects in dimension H^1. The divisive clustering starts at the top of one big cluster and at each level recursively splits one of existing clusters into two new clusters to decrease the distance. Like the induction of decision tree, for example, we can select an attribute-value pair (H^m,h^m) to split the cluster into two, depending on the new objects' m-th attributes. The advantage of these hierarchical methods is that they provide a chance to explicitly check the objective function. The disadvantage is the computational cost, which may not be a problem for a set of similar candidates.

The above joint-cluster is a special case of micro-cluster. With a predetermined threshold ϵ, a micro-cluster refers to any subset of data that are assigned to the same cluster by at least a fraction ϵ of candidate clusterings. Thus a joint-cluster is a micro-cluster with full support 1. By treating original data as items and each original cluster in candidates as a transaction, frequent itemsets can be mined and they are used to construct a weighted hypergraph. Each frequent itemset is a hyperedge whose weight is its support, i.e., the fraction of candidate clusterings that assign all data in the itemset together. Then the hypergraph partitioning algorithm hMETIS [79] can be employed to partition the constructed hypergraph. Similar idea appears in [51] for clustering customer transactions in a market basket database. Their goal is to use the result from hMETIS, a clustering of items, to partition the transactions. The data in their problem are very different from ours. A customer transaction often contains a small number of items in contrast to the huge total item size. In our case, a cluster usually contains a considerable fraction of total data.

If all candidate clusterings have approximately the same number of clusters, we can assume that there is a one-to-one mapping between clusters in different candidates. The similarity between two clusters in two different candidates respectively can be computed with the binary Jaccard coefficient, i.e., the size ratio of their join over their union. Then the pool of all clusters can be partitioned by METIS. The resulting cluster is called macro-cluster, because it contains a few original clusters in different candidates. Some score function is used to assign data to the closest macro-cluster, e.g., using the number of occurences of data in all original clusters contained in the macro-cluster. This is similar to multi-clustering fusion methods presented in [31, 36], where evidence is accumulated based on combining intermediate results from an iterative clustering algorithm.

6.2.5 Finding Outliers: An Information Theory Perspective

From information theory perspective, an outlier can be regarded as the one with more information, surprise, etc. Intuitively, given a dataset D and a new data point x, the outlier factor of x can be defined as $-\log(P(x|D))$, i.e., it is more outlying (surprising) if the probability to predict it given D is smaller. Often we assume some class of parametric distribution h for D and try to estimate $P(x|h, D)$. But h is after all imaginary and it is hard to verify if it is the true distribution governing the whole population. First, it is difficult to justify our model selection, say, Gaussian over exponential. Besides, even within the same class such as Gaussian mixture, it is much harder to determine the number of components than the parameters inside each component [70]. In other words, model selection is much harder than parameter estimation. This problem of selection of the imaginary distribution (model) can be partially circumvented by looking at it from another perspective of information criteria, such as Akaike's information criterion [3], minimum description length (MDL) [8] or stochastic complexity [109]. Generally, MDL tries to represent an entire class of probability distributions as models by a single universal representative model so that we are able to imitate the behavior of any model in the class. Information contained by a dataset can be measured in terms of the code length and the outlier factor can be measured in terms of the increase in the code length after incorporating this new data point. Although this principle seems simple, it offers a fundamental change in the way we model data, for we need not assume that they are from an imagined distribution. According to this program, the problems of modeling and inference no longer has to be estimating any true data generating distribution on which we base the inference, but to search for good models for the data, where the goodness is measured in terms of code length.

Therefore, if we want to find a universal approach to detecting all kinds of outliers

with some theoretical justification, information theory is a good starting point. But is it too ambitious? In a way, all work involved in pattern recognition, machine learning and even statistics is nothing more than summarizing and modeling data and making statistical inference. Perhaps it is more practical to first construct approaches for each pattern separately. This is supported by Vapnik's philosophy in his milestone book on statistical learning theory [125]: 'If you possess a restricted amount of information for solving some problem, try to solve the problem directly and never solve a more general problem as an intermediate step. It is possible that the available information is sufficient for a direct solution but is insufficient for solving a more general intermediate problem.'

Bibliography

[1] C. C. Aggarwal and P. S. Yu. Outlier detection for high dimensional data. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, pages 37–46, 2001.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 94 – 105, 1998.

[3] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[4] C. Ambroise and G. Govaert. Convergence of an EM-type algorithm for spatial clustering. *Pattern Recognition Letters*, 19(10):919 – 927, 1998.

[5] M. Ankerst, M. M. Breunig, H. P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. In *Proceedings of 1999 ACM SIGMOD International Conference on Management of Data*, pages 49–60, 1999.

[6] L. Anselin. *Spatial Econometrics: Methods and Models*. Kluwer Academic Publishers, 1988.

[7] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley & Sons, 3 edition, 1994.

[8] A. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760, 1998.

[9] R. J. Beckman and R. D. Cook. Outliers. *Technometrics*, 25:119–149, 1983.

[10] S. Berchthold, D. A. Keim, and H. P. Kriegel. The x-tree: An index structure for high-dimensional data. In *Proceedings of The 22nd International Conference on Very Large Data Bases*, pages 28–39, 1996.

[11] J. E. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, B(36):192–225, 1974.

[12] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[13] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *Proceedings of 7th IEEE International Conference on Computer Vision*, pages 377–384, 1999.

[14] P.S. Bradley and U. M. Fayyad. Refining initial points for k–means clustering. In *Proceedings of the 14th International Conference on Machine Learning*, pages 91–99, 1998.

[15] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 93–104, 2000.

[16] W. Buntine and T. Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 8:75–86, 1992.

[17] A. De Carvalho and M. M. Brizzotti. Combining RBF networks trained by different clustering techniques. *Neural Processing Letters*, 14:227–240, 2001.

[18] G. Celeux and G. Govaert. A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14(3):315–332, 1992.

[19] S. Chen, C. F. N. Cowan, and P. M. Grant. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 2(2):302–309, 1991.

[20] N. A. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, revised edition, 1993.

[21] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society*, B(39):1–38, 1977.

[22] T. G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the 2nd International Workshop on Multiple Classifier Systems*, pages 1–15, 2001.

[23] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

[24] H. Edelsbrunner. *Algorithms in Computational Geometry*. Springer-Verlag, 1987.

[25] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[26] M. Ester, H.P. Kriegel, and J. Sander. Spatial data mining: A database approach. In *Proceedings of 5th Symposium on Spatial Databases*, pages 47–66, 1997.

[27] V. Estivill-Castro and I. Lee. Fast spatial clustering with different metrics and in the presence of obstacles. In *Proceedings of the 9th ACM International Symposium on Advances in Geographic Information Systems*, pages 142 – 147, 2001.

[28] U. Fayyad, D. Haussler, and P. Stolorz. Mining scientific data. *Communications of the ACM*, 39(11):51–57, 1996.

[29] U. M. Fayyad, C. Reina, and P. S. Bradley. Initialization of iterative refinement clustering algorithms. In *Proceedings of the 14th International Conference on Machine Learning*, pages 194–198, 1998.

[30] D. Fisher. Iterative optimization and simplification of hierarchical clusterings. *Journal of Artificial Intelligence Research*, 4:147–180, 1996.

[31] A. L. N. Fred and A. K. Jain. Evidence accumulation clustering based on the k-means algorithm. In *Proceedings of the Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition*, pages 442–451, 2002.

[32] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, pages 148–156, 1996.

[33] J. H. Friedman. Greedy function approximation: The gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001.

[34] J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28(2):337–374, 2000.

[35] D. Frossyniotis, A. Likas, and A. Stafylopatis. A clustering method based on boosting. *Pattern Recognition Letters*, 25(6):641–654, 2004.

[36] D. Frossyniotis, M. Pertselakis, and A. Stafylopatis. A multi-clustering fusion algorithm. In *Proceedings of the 2nd Hellenic Conference on Artificial Intelligence*, pages 225–236, 2002.

[37] V. Ganti, J. Gehrke, and R. Ramakrishnan. Cactus-clustering categorical data using summaries. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 73–83, 1999.

[38] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[39] J. Ghosh. Multiclassifier systems: Back to the future. In *Proceedings of the 3rd International Workshop on Multiple Classifier Systems*, pages 1–15, 2002.

[40] N. Gilardi and S. Bengio. Local machine learning models for spatial data analysis. *Journal of Geographic Information and Decision Analysis*, 4(1):11–28, 2000.

[41] O. W. Gilley and R. K. Pace. On the harrison and rubinfeld data. *Journal of Environmental Economics and Management*, 31:403–405, 1996.

[42] A. Gordon. *Classification*. Chapman and Hall / CRC Press, 2nd edition, 1999.

[43] J. Grabmeier and A. Rudolph. Techniques of cluster algorithms in data mining. *Data Mining and Knowledge Discovery*, 6(4):303–360, 2002.

[44] D. Griffith. Statistical and mathematical sources of regional science theory: Map pattern analysis as an example. *Papers in Regional Science*, 78:21–45, 1999.

[45] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 73–84, 1998.

[46] S. Guha, R. Rastogi, and K. Shim. Rock: A robust clustering algorithm for categorical attributes. In *Proceedings of 15th International Conference on Data Engineering*, pages 512–521, 1999.

[47] D. Guo, D. Peuquet, and M. Gahegan. Opening the black box: Interactive hierarchical clustering for multivariate spatial patterns. In *Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems*, pages 131 – 136, 2002.

[48] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, pages 47–57, 1984.

[49] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Cluster validity methods: Part I. *SIGMOD Record*, 31(2):40–45, 2002.

[50] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. Clustering validity checking methods: Part II. *SIGMOD Record*, 31(3):19–27, 2002.

[51] E. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs. In *Proceedings of the 1997 ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, 1997.

[52] D. J. Hand. *Discrimination and Classification*. John Wiley & Sons, 1981.

[53] D. Harel and Y. Koren. Clustering spatial data using random walks. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 281–286, 2001.

[54] D. Harrison and D. L. Rubinfeld. Hedonic prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5:81–102, 1978.

[55] E. J. Hartman, J. D. Keller, and J. M. Kowalski. Layered neural networks with gaussian hidden units as universal approximations. *Neural Computation*, 2(2):210–215, 1990.

[56] T. Hastie, R. Tibshirani, and J. Friedman. *Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer-Verlag, 2001.

[57] R. J. Hathaway. Another interpretation of the EM algorithm for mixture distributions. *Statistics and Probability Letters*, 4:53–56, 1986.

[58] D. M. Hawkins. *Identification of Outliers*. Chapman and Hall, 1980.

[59] L. Hermes and J. M. Buhmann. Contextual classification by entropy-based polygonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 442–447, 2001.

[60] A. Hinneburg and D. A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[61] T. Hu and S. Y. Sung. Detecting pattern-based outliers. *Pattern Recognition Letters*, 24(16):3059 – 3068, 2003.

[62] T. Hu and S. Y. Sung. Spatial similarity measures in location prediction. *Journal of Geographic Information and Decision Analysis*, 7(2):93–104, 2003.

[63] T. Hu and S. Y. Sung. A hybrid EM approach to spatial clustering. Accepted by Computational Statistics and Data Analysis, 2004.

[64] T. Hu and S. Y. Sung. A trimmed mean approach to finding spatial outliers. *Intelligent Data Analysis*, 8(1):79–95, 2004.

[65] T. Hu and S. Y. Sung. Data fusion in radial basis function network for spatial regression. *Neural Processing Letters*, 21(2):81 – 93, 2005.

[66] T. Hu and S. Y. Sung. Finding centroid clusterings with entropy-based criteria. Accepted by Knowledge and Information Systems, 2005.

[67] T. Hu and S. Y. Sung. Finding outliers at multiple scales. *International Journal of Information Technology and Decision Making*, 4(2):251–262, 2005.

[68] L. J. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:63–76, 1985.

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

[70] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[71] A. K. Jain and F. Farrokhnia. Unsupervised texture segmentation using Gabor filters. *Pattern Recognition*, 24(12):1167–1186, 1991.

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

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

[74] E. Johnson and H. Kargupta. Collective, hierarchical clustering from distributed, heterogeneous data. In *Large-Scale Parallel KDD Systems*, pages 221–244. Springer-Verlag, 1999.

[75] T. Johnson, I. Kwok, and R.T. Ng. Fast computation of 2-dimensional depth contours. In *Proceedings of the 4th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 224–228, 1998.

[76] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

[77] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of AI Research*, 4:237–285, 1996.

[78] H. Kargupta, W. Huang, and E. Johnson. Distributed clustering using collective principal component analysis. *Knowledge and Information Systems Journal*, 3:422–448, 2001.

[79] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Applications in VLSI domain. In *Proceedings of the 34th Conference on Design Automation*, pages 526–529, 1997.

[80] G. Karypis, E. H. Han, and V. Kumar. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *Computer*, 32(8):68–75, 1999.

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

[82] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, 1990.

[83] E. M. Knorr and R. T. Ng. Finding aggregate proximity relationships and commonalities in spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):884–897, 1996.

[84] E. M. Knorr and R. T. Ng. Finding intensional knowledge of distance-based outliers. In *Proceedings of the 25th International Conference on Very Large Data Bases*, pages 211–222, 1999.

[85] E. M. Knorr, R. T. Ng, and V. Tucakov. Distance-based outliers: Algorithms and applications. *The Very Large Data Bases Journal*, 8(3):237–253, 2000.

[86] K. Koperski, J. Adhikary, and J. Han. Spatial data mining: Progress and challenges. In *Proceedings of Workshop Research Issues on Data Mining and Knowledge Discovery*, pages 1–10, 1996.

[87] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[88] P. Legendre. Constrained clustering. In P. Legendre and L. Legendre, editors, *Developments in Numerical Ecology*, pages 289–307, 1987. NATO ASI Series G 14.

[89] J. P. LeSage. Bayesian estimation of spatial autoregressive models. *International Regional Science Review*, 20:113–129, 1997.

[90] J. P. LeSage. *MATLAB Toolbox for Spatial Econometrics*. http://www.spatial-econometrics.com, 1999.

[91] H. Leung, G. Hennessey, and A. Drosopoulos. Signal detection using the radial basis function coupled map lattice. *IEEE Transactions on Neural Networks*, 11(5):1133 –1151, 2000.

[92] B. Liu, W. Hsu, L. Mun, and H. Lee. Finding interesting patterns using user expectations. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):817–832, 1999.

[93] R. Mceliece. *Theory of Information and Coding.* Addison-Wesley, 1977.

[94] M. Mehrotra. Multi-viewpoint clustering analysis (mvp-ca) technology for mission rule set development and case-based retrieval. Technical Report AFRL-VS-TR-1999-1029, Air Force Research Laboratory, 1999.

[95] P. Michaud. Condorcet - a man of the avant-garde. *Applied Stochastic Models and Data Analysis*, 3:173–198, 1987.

[96] T. M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[97] P. M. Murphy and D. W. Aha. *UCI Repository of Machine Learning Databases.* Department of Information and Computer Science, University of California at Irvine, 1994. http://www.ics.uci.edu/ $\sim$ mlearn/MLRepository.html.

[98] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[99] C. Neukirchen, J. Rottland, D. Willett, and G. Rigoll. A continuous density interpretation of discrete HMM systems and MMI-neural networks. *IEEE Transactions on Speech and Audio Processing*, 9(4):367–377, 2001.

[100] R. Ng and J. Han. CLARANS: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):1003–1016, 2002.

[101] M. A. Oliver and R. Webster. A geostatistical basis for spatial weighting in multivariate classification. *Mathematical Geology*, 21:15–35, 1989.

[102] R. K. Pace and R. Barry. Quick computation of spatial autoregressive estimators. *Geographical Analysis*, 29:232–247, 1997.

[103] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[104] M. J. D. Powell. Radial basis functions for multivariable interpolation: A review. In *Algorithms for Approximation*, pages 143–167. Oxford: Clarendon Press, 1987.

[105] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[106] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 427–438, 2000.

[107] J. P. Rasson and V. Granville. Multivariate discriminant analysis and maximum penalized likelihood density estimation. *Journal of the Royal Statistical Society*, B(57):501–517, 1995.

[108] G. Rigoll. Maximum mutual information neural networks for hybrid connectionist-HMM speech recognition systems. *IEEE Transactions on Speech and Audio Processing*, 2(1):175–184, 1994.

[109] J. J. Rissanen. Fisher information and stochastic complexity. *IEEE Transactions on Information Theory*, 42(1):40 –47, 1996.

[110] J. F. Roddick and M. Spiliopoulou. A bibliography of temporal, spatial and spatio-temporal data mining research. *ACM SIGKDD Explorations*, 1(1):34–38, 1999.

[111] S. Ross. *A First Course in Probability*. Prentice Hall, 5th edition, 1998.

[112] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 71–79, 1995.

[113] I. Ruts and P. Rousseeuw. Computing depth contours of bivariate point clouds. *Computational Statistics and Data Analysis*, 23:153–168, 1996.

[114] R. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

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

[116] G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. In *Proceedings of the 24th International Conference on Very Large Data Bases*, pages 428–439, 1998.

[117] S. Shekhar and S. Chawla. *Spatial Databases: A Tour*. Prentice-Hall, 2002.

[118] S. Shekhar, P. Schrater, W. R. Raju, W. Wu, and S. Chawla. Spatial contextual classification and prediction models for mining geospatial data. *IEEE Transactions on Multimedia*, 4(2):174–188, 2002.

[119] A. Silberschatz and A. Tuchilin. What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):970–974, 1996.

[120] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[121] A. H. Solberg, T. Taxt, and A. K. Jain. A markov random field model for classification of multisource satellite imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 34(1):100–113, 1996.

[122] W. R. Tobler. *Cellular Geography, Philosophy in Geography*. Dordrecht, Reidel, 1979.

[123] A. K. H. Tung, J. Hou, and J. Han. Spatial clustering in the presence of obstacles. In *Proceedings of 17th International Conference on Data Engineering*, pages 359–367, 2001.

[124] L. Valiant. A theory of learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[125] V. N. Vapnik. *Statistical Learning Theory*. New York: John Wiley & Sons, 1998.

[126] W. Wang, J. Yang, and R. Muntz. STING: A statistical information grid approach to spatial data mining. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 186–195, 1997.

[127] R. Williams and D. Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. In *Backpropagation: Theory, Architectures, and Applications*, pages 433–486. Lawrence Erlbaum Associates, Hillsdale, NJ, 1995.

[128] L. Xu and M. I. Jordan. On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Computation*, 8:129–151, 1996.

[129] X. Xu, M. Ester, H. P. Kriegel, and J. Sander. A distribution-based clustering algorithm for mining in large spatial databases. In *Proceedings of 14th International Conference on Data Engineering*, pages 324–331, 1998.

[130] K. Yamanishi, J. Takeuchi, G. Williams, and P. Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 320 – 324, 2000.

[131] Y. Yan. Understanding speech recognition using correlation-generated neural network targets. *IEEE Transactions on Speech and Audio Processing*, 7(3):350–352, 1999.

[132] H. Yin and N. M. Allinson. Self-organizing mixture networks for probability density estimation. *IEEE Transactions on Neural Networks*, 12(2):405–411, 2001.

[133] T. Zhang, R. Ramakrishnan, and M. Linvy. BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 103–114, 1996.

[134] Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *Proceedings of the 2002 ACM CIKM International Conference on Information and Knowledge Management*, pages 515–524, 2002.

Appendix A

Proof of Triangle Inequality

We give two proofs, the first purely based on inequality manipulation, the second using decomposition with more descriptive flavor.

A.1 Proof by Manipulation

Triangle inequality in Eq. (4.6) is equivalent to

$$
\begin{aligned}
& d(Y,Z) - d(X,Y) - d(X,Z) \leq 0 \\
\Leftrightarrow \quad & H(Y,Z) - H(X,Z) - (H(X,Y) - H(X)) \leq 0 \qquad \text{(A.1)} \\
\Leftrightarrow \quad & (H(Y,Z) - H(Z)) - (H(X,Z) - H(Z)) - (H(X,Y) - H(X)) \leq 0 \\
\Leftrightarrow \quad & H(Y|Z) - H(X|Z) - H(Y|X) \leq 0 \qquad \text{(A.2)}
\end{aligned}
$$

where Eq. (A.1) is derived using Eq. (4.4). Before proving Eq. (A.2), we need the following lemma: $\forall x > 0, \ln x \leq x - 1$, with equality only at $x = 1$. Its proof is very simple by comparing derivatives.

Assuming X, Y and Z can take on values in $\{x_i\}$, $\{y_j\}$ and $\{z_k\}$, respectively, we have

$$
\begin{aligned}
& H(Y|Z) - H(X|Z) - H(Y|X) \\
= & -\sum_k \sum_j p(z_k)p(y_j|z_k)\ln p(y_j|z_k) + \sum_k \sum_i p(z_k)p(x_i|z_k)\ln p(x_i|z_k) \\
& + \sum_i \sum_j p(x_i)p(y_j|x_i)\ln p(y_j|x_i) \\
= & -\sum_k \sum_j p(y_j, z_k)\ln p(y_j|z_k) + \sum_k \sum_i p(x_i, z_k)\ln p(x_i|z_k) + \sum_i \sum_j p(x_i, y_j)\ln p(y_j|x_i) \\
= & -\sum_i \sum_j \sum_k p(x_i, y_j, z_k)\ln p(y_j|z_k) + \sum_i \sum_j \sum_k p(x_i, y_j, z_k)\ln p(x_i|z_k) \\
& + \sum_i \sum_j \sum_k p(x_i, y_j, z_k)\ln p(y_j|x_i) \\
= & \sum_i \sum_j \sum_k p(x_i, y_j, z_k)\ln\left[\frac{p(x_i|z_k)p(y_j|x_i)}{p(y_j|z_k)}\right] \\
\leq & \sum_i \sum_j \sum_k p(x_i, y_j, z_k)\left[\frac{p(x_i|z_k)p(y_j|x_i)}{p(y_j|z_k)} - 1\right] \\
= & \sum_i \sum_j \sum_k p(z_k)p(y_j|z_k)p(x_i|y_j, z_k)\frac{p(x_i|z_k)p(y_j|x_i)}{p(y_j|z_k)} - 1 \\
= & \sum_i \sum_j \sum_k p(z_k)p(x_i|y_j, z_k)p(x_i|z_k)p(y_j|x_i) - 1 \\
= & \sum_i \sum_j \sum_k p(x_i|y_j, z_k)p(x_i, z_k)p(y_j|x_i) - 1 \\
\leq & \sum_i \sum_j \sum_k p(x_i, z_k)p(y_j|x_i) - 1 \\
= & \sum_i \sum_j p(x_i)p(y_j|x_i) - 1 \\
= & \sum_i \sum_j p(x_i, y_j) - 1 \\
= & 0
\end{aligned}
$$

A.2 Proof by Decomposition

Triangle inequality in Eq. (4.6) is equivalent to

$$H(X) + H(Y, Z) \leq H(X, Y) + H(X, Z) \tag{A.3}$$

If X is a single cluster or $H(X) = 0$, then $H(X, Y) = H(Y)$ and $H(X, Z) = H(Z)$.

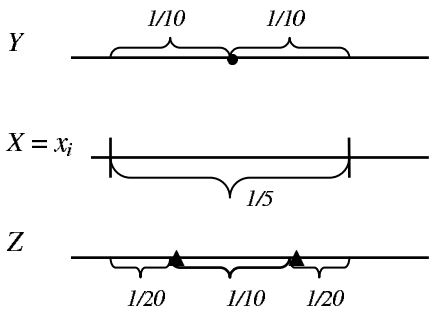

Figure A.1: Data of cluster x_i ($p(x_i) = 1/5$) in clustering X are distributed into two clusters in clustering Y and three clusters in clustering Z, respectively.

From Eq. (4.1) we have Eq. (A.3) is true in this case.

If X contains more than one cluster, again we assume X, Y and Z take on values in $\{x_{i'}\}$, $\{y_j\}$ and $\{z_k\}$, respectively. First we restrict our discussion on one particular cluster x_i with an illustrative example in Fig. A.1, where data in x_i ($p(x_i) = 1/5$) are distributed into two clusters in Y and three clusters in Z, respectively. When restricted to cluster x_i of X, we can decompose $H(X)$ as

$$H(X) = \sum_{i'} p(x_{i'}) \ln[1/p(x_{i'})] = ... + \frac{\ln 5}{5} + ...$$

Note that $\frac{\ln 5}{5}$ is the summand corresponding to cluster x_i, which can be denoted by $H(X)|_{X=x_i}$. Similarly, other terms in Eq. (A.3) can be decomposed as

$$\begin{aligned}
H(Y,Z) &\leq H(X,Y,Z) = \sum_{i'} \left[\sum_{j,k} p(x_{i'}, y_j, z_k) \ln[1/p(x_{i'}, y_j, z_k)] \right] = ... + \frac{\ln 20}{5} + ... \\
H(X,Y) &= \sum_{i'} \left[\sum_{j} p(x_{i'}, y_j) \ln[1/p(x_{i'}, y_j)] \right] = ... + \frac{\ln 10}{5} + ... \\
H(X,Z) &= \sum_{i'} \left[\sum_{k} p(x_{i'}, z_k) \ln[1/p(x_{i'}, z_k)] \right] = ... + (\frac{\ln 10}{10} + \frac{\ln 20}{10}) + ...
\end{aligned}$$

It is easy to check that when $X = x_i$ Eq. (A.3) is true for the corresponding components, namely

$$[H(X) + H(X,Y,Z)]|_{X=x_i} \leq [H(X,Y) + H(X,Z)]|_{X=x_i} \quad \text{(A.4)}$$

since the left side is equal to $\frac{2\ln 5}{5} + \frac{2\ln 2}{5}$ and the right side is equal to $\frac{2\ln 5}{5} + \frac{\ln 2}{2}$. Eq. (A.3) is proved if we can prove the above relation for every component for the general case. Suppose that the cluster x_i in X under examination has probability $p(x_i) = 1/a$. Then the corresponding components in every term of Eq. (A.3) can be written as

$$\begin{aligned}
H(X)|_{X=x_i} &= \frac{\ln a}{a} \\
H(Y,Z)|_{X=x_i} &\leq H(X,Y,Z)|_{X=x_i} = \sum_l q_l \ln(\frac{1}{q_l}),\ \sum_l q_l = \frac{1}{a} \\
H(X,Y)|_{X=x_i} &= \sum_m r_m \ln(\frac{1}{r_m}),\ \sum_m r_m = \frac{1}{a} \\
H(X,Z)|_{X=x_i} &= \sum_n s_n \ln(\frac{1}{s_n}),\ \sum_n s_n = \frac{1}{a}
\end{aligned}$$

where we use $\{q_l\}, \{r_m\}$ and $\{s_n\}$ to denote the distribution of data of x_i in other clusterings. For instance, in the above example in Fig. A.1, $\{s_n\} = \{1/20, 1/10, 1/20\}$. By adding $2\frac{1}{a}\ln\frac{1}{a}$ to both sides of Eq. (A.4) for the general case, we have

$$\begin{aligned}
&[H(X) + H(X,Y,Z)]|_{X=x_i} + 2\frac{1}{a}\ln\frac{1}{a} \qquad [H(X,Y) + H(X,Z)]|_{X=x_i} + 2\frac{1}{a}\ln\frac{1}{a} \\
=&\frac{\ln a}{a} + \sum_l q_l \ln\frac{1}{q_l} + 2\frac{1}{a}\ln\frac{1}{a} \qquad = \sum_m r_m \ln\frac{1}{r_m} + \sum_n s_n \ln\frac{1}{s_n} + 2\frac{1}{a}\ln\frac{1}{a} \\
=&\sum_l q_l \ln\frac{1}{aq_l} \qquad = \sum_m r_m \ln\frac{1}{ar_m} + \sum_n s_n \ln\frac{1}{as_n} \\
=&\frac{1}{a}\sum_l aq_l \ln\frac{1}{aq_l} \qquad = \frac{1}{a}\left(\sum_m ar_m \ln\frac{1}{ar_m} + \sum_n as_n \ln\frac{1}{as_n}\right)
\end{aligned}$$

Note that $\sum_n as_n = 1$ and hence $\sum_n as_n \ln\frac{1}{as_n}$ is the entropy of a certain distribution. In fact this distribution is none other than the conditional distribution/clustering $(Z|X = x_i)$. For instance, in the above example in Fig. A.1, $\{as_n\} = \{1/4, 1/2, 1/4\}$.

Similarly, $\sum_m ar_m \ln\frac{1}{ar_m}$ and $\sum_l aq_l \ln\frac{1}{aq_l}$ correspond to $(Y|X=x_i)$ and $(Y,Z|X=x_i)$, respectively. For the example in Fig. A.1, these entropies are $H(Y|X=x_i) = \ln 2, H(Z|X=x_i) = \frac{3}{2}\ln 2, H(Y,Z|X=x_i) = 2\ln 2$. Therefore, from Eq. (4.1), we have

$$\sum_l aq_l \ln\frac{1}{aq_l} \leq \sum_m ar_m \ln\frac{1}{ar_m} + \sum_n as_n \ln\frac{1}{as_n}$$

which means that Eq. (A.4) is true. Similarly, it is also true for every other cluster of X and thus Eq. (A.3) is proved.