NONPARAMETRIC BAYESIAN MODELS FOR UNSUPERVISED LEARNING

by

Pu Wang

A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Computer Science

Committee:

Carlotta Domeniconi, Dissertation Director

Kathryn B. Laskey, Committee Member

Jana Kosecka, Committee Member

Huzefa Rangwala, Committee Member

Hassan Gomaa, Department Chair

Lloyd J. Griffiths, Dean, The Volgenau School of Engineering

Date: 5/4/11

Spring 2011
George Mason University
Fairfax, VA

Nonparametric Bayesian Models for Unsupervised Learning

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Pu Wang
Master of Science
Peking University, 2007
Bachelor of Engineering
Beihang University, 2004

Director: Carlotta Domeniconi, Professor
Department of Computer Science

Spring 2011
George Mason University
Fairfax, VA

Dedication

I dedicate this dissertation to my parents.

Acknowledgments

I would like to thank the following people who made this possible, my advisors, and my committee members. Without their guidance, I would never be able to finish my PhD work.

First, I would like to express my deepest gratitude to my two advisors, Prof. Carlotta Domeniconi, and Prof. Kathryn B. Laskey. Prof. Domeniconi introduced me to the research area of machine learning, and gave me the environment and guidance to do research. Prof. Laskey guided me through the challenges and difficulties of Bayesian statistics. Further, both Prof. Domeniconi and Prof. Laskey helped me to develop crucial research-related abilities, writing and presenting, which greatly increased my research capability.

I am very grateful to the two other members in my committee, Prof. Jana Kosecka and Prof. Huzefa Rangwala. Prof. Kosecka helped me understand computer vision problems, and convinced me that a researcher should be more data-driven than model-driven. Prof. Rangwala taught me bioinformatics, and showed me how to conduct comprehensive experiments. I obtained valuable research experience from them, and with their help I learned how to apply my knowledge to solve practical problems.

Also, I am very thankful to Prof. Sean Luke. Prof. Luke kept giving me advice on programming, engineering and other aspects, such American culture. It is Prof. Luke who showed me how to be a good researcher and a good engineer at the same time.

Additionally, I must thank all the members of the Data Mining lab and all the participants in the Krypton seminar. They gave me a lot of valuable comments on my dissertation and defense.

Finally, I would like to thank my parents. They always supported me and encouraged me with their best wishes.

Table of Contents

List of Tables

List of Figures

Abstract

NONPARAMETRIC BAYESIAN MODELS FOR UNSUPERVISED LEARNING

Pu Wang, PhD

George Mason University, 2011

Dissertation Director: Carlotta Domeniconi

Unsupervised learning is an important topic in machine learning. In particular, clustering is an unsupervised learning problem that arises in a variety of applications for data analysis and mining. Unfortunately, clustering is an ill-posed problem and, as such, a challenging one: no ground-truth that can be used to validate clustering results is available. Two issues arise as a consequence. Various clustering algorithms embed their own bias resulting from different optimization criteria. As a result, each algorithm may discover different patterns in a given dataset. The second issue concerns the setting of parameters. In clustering, parameter setting controls the characterization of individual clusters, and the total number of clusters in the data.

Clustering ensembles have been proposed to address the issue of different biases induced by various algorithms. Clustering ensembles combine different clustering results, and can provide solutions that are robust against spurious elements in the data. Although clustering ensembles provide a significant advance, they do not address satisfactorily the model selection and the parameter tuning problem.

Bayesian approaches have been applied to clustering to address the parameter tuning and model selection issues. Bayesian methods provide a principled way to address these problems by assuming prior distributions on model parameters. Prior distributions assign low probabilities to parameter values which are unlikely. Therefore they serve as regularizers for modeling parameters, and can help avoid over-fitting. In addition, the marginal likelihood is used by Bayesian approaches as the criterion for model selection. Although Bayesian methods provide a principled way to perform parameter tuning and model selection, the key question "How many clusters?" is still open. This is a fundamental question for model selection. A special kind of Bayesian methods, nonparametric Bayesian approaches, have been proposed to address this important model selection issue. Unlike parametric Bayesian models, for which the number of parameters is finite and fixed, nonparametric Bayesian models allow the number of parameters to grow with the number of observations. After observing the data, nonparametric Bayesian models fit the data with finite dimensional parameters.

An additional issue with clustering is high dimensionality. High-dimensional data pose a difficult challenge to the clustering process. A common scenario with high-dimensional data is that clusters may exist in different subspaces comprised of different combinations of features (dimensions). In other words, data points in a cluster may be similar to each other along a subset of dimensions, but not in all dimensions. People have proposed *subspace clustering* techniques, a.k.a. *co-clustering* or *bi-clustering*, to address the dimensionality issue (here, I use the term co-clustering). Like clustering, also co-clustering suffers from the ill-posed nature and the lack of ground-truth to validate the results.

Although attempts have been made in the literature to address individually the major issues related to clustering, no previous work has addressed them jointly. In my dissertation I propose a unified framework that addresses all three issues at the same time. I designed a nonparametric Bayesian clustering ensemble (NBCE) approach, which assumes that multiple observed clustering results are generated from an unknown consensus clustering. The underlying distribution is assumed to be a mixture distribution with a nonparametric Bayesian prior, i.e., a Dirichlet Process. The number of mixture components, a.k.a. the number of consensus clusters, is learned automatically. By combining the ensemble methodology and nonparametric Bayesian modeling, NBCE addresses both the ill-posed nature and the parameter setting/model selection issues of clustering. Furthermore, NBCE outperforms individual clustering methods, since it can escape local optima by combining multiple clustering results.

I also designed a nonparametric Bayesian co-clustering ensemble (NBCCE) technique. NBCCE inherits the advantages of NBCE, and in addition it is effective with high dimensional data. As such, NBCCE provides a unified framework to address all the three aforementioned issues. NBCCE assumes that multiple observed co-clustering results are generated from an unknown consensus co-clustering. The underlying distribution is assumed to be a mixture with a nonparametric Bayesian prior. I developed two models to generate co-clusters in terms of row- and column- clusters. In one case row- and column-clusters are assumed to be independent, and NBCCE assumes two independent Dirichlet Process priors on the hidden consensus co-clustering, one for rows and one for columns. The second model captures the dependence between row- and column-clusters by assuming a Mondrian Process prior on the hidden consensus co-clustering. Combined with Mondrian priors, NBCCE provides more flexibility to fit the data.

I have performed extensive evaluation on relational data and protein-molecule interaction data. The empirical evaluation demonstrates the effectiveness of NBCE and NBCCE and their advantages over traditional clustering and co-clustering methods.

Chapter 1: Introduction

1.1 Motivation

Unsupervised learning is an important topic in machine learning. In particular, clustering is an unsupervised learning problem that arises in a variety of applications for data analysis and mining. The aim of clustering is to organize data into groups so that points similar to each other are placed in the same cluster and points different from one another are placed in different clusters. For example, one can cluster documents according to content. Documents with similar content will have high similarity, and they are more likely to be clustered together.

Unfortunately, clustering is an ill-posed problem and, as such, a challenging one: no ground-truth that can be used to validate clustering results is available. Two issues arise as a consequence. Various clustering algorithms embed their own bias resulting from different optimization criteria. As a result, each algorithm may discover different patterns in a given dataset. The second issue concerns the setting of parameters and model selection. Model selection is to select a model from a set of candidate models. Each candidate model is characterized by some parameters. In clustering, parameter setting and model selection involves at least two aspects, namely the characterization of individual clusters, and the total number of clusters in the data. Due to the absence of ground-truth, cross-validation techniques cannot be used to tune the involved input parameters. As a consequence, model selection becomes challenging for clustering: users have no guidelines for choosing the proper clustering model for a given dataset. Here I refer to the two issues as different bias, and model selection and parameter setting.

Clustering ensembles have been proposed to address the issue of different biases induced by various algorithms. Clustering ensembles combine different clustering results. Different

clusterings can be obtained from clustering the same dataset w.r.t. different criteria, or from different local optima of the same clustering algorithm obtained using different parameter values on the same dataset. Clustering ensembles can provide solutions that are robust against spurious elements in the data. By combining multiple clustering results, the combination process allows to cancel out emergent spurious structures that arise due to the various biases to which each individual clustering is tuned, or to the variance induced by different local optima. The clustering result of a clustering ensemble is called *consensus clustering*.

Although clustering ensembles provide a significant advance, they do not address satisfactorily the model selection and the parameter setting problem. For example, although clustering ensembles can combine clustering results of varying number of clusters, they still need users to specify the number of consensus clusters. Therefore it's still challenging for clustering ensembles to perform model selection and parameter tuning, due to the absence of ground-truth.

Bayesian approaches have been applied to clustering to address the parameter tuning and model selection issues. Bayesian methods provide a principled way to address these problems by assuming prior distributions on model parameters. For example, when using mixture models for clustering, each mixture component is considered as a cluster, and Bayesian mixture models assume prior distributions to the parameters of each mixture component and to the weights of mixture components. Prior distributions assign low probabilities to parameter values which are unlikely. Therefore they serve as regularizers for modeling parameters, and can help avoid over-fitting. In addition, the marginal likelihood is used by Bayesian approaches as the criterion for model selection. The marginal likelihood is defined as $p(D|m) = \int p(D|\theta, m)p(\theta|m)d\theta$, where D denotes the observed data, m is a specific model, and θ represents the model parameters. The marginal likelihood can be interpreted as the probability of the data under the model, averaging over all possible parameter values. In general, we want to avoid using too simple or too complex models, since both too simple and too complex models do not generalize well on unseen data. If the model is too simple, it is unlikely to generate the observed data D; on the other hand, if the model is too complex,

it can generate many possible data, therefore it is unlikely to generate that particular data D at random. This is the built-in property of Bayesian methods when performing model selection, called *Occam's Razor* [46]. When predicting using Bayesian methods, the predictive distribution is $p(\vec{x}|D,m) = \int p(\vec{x}|\theta,D,m)p(\theta|D,m)d\theta$, where $\vec{x}$ is an unseen datum. The predictive distribution demonstrates the key ingredient of Bayesian methods, the averaging over uncertain variables and parameters. This means that Bayesian methods do not perform a point estimation $\hat{\theta}$ of the model parameter θ to predict unseen data, but learn a posterior distribution of the model parameter, $p(\theta|D,m)$, and then predict unseen data using the expectation of unseen data w.r.t. the posterior of the model parameter.

Although Bayesian methods provide a principled way to perform parameter tuning and model selection, the key question "How many clusters?" is still open. This is a fundamental question for model selection. Specifically it relates to how many parameters should be used in clustering models. For example, if we use a Bayesian mixture model for clustering, we can assume there are K mixture components, and therefore we need K parameters to represent the K components (or clusters). A special kind of Bayesian methods, *nonparametric Bayesian approaches*, have been proposed to address this important model selection issue. Unlike parametric Bayesian models, for which the number of parameters is finite and fixed, nonparametric Bayesian models allow the number of parameters to grow with the number of observations. To accommodate asymptotically unbounded numbers of parameters within a single parameter space, the dimension of the space has to be infinite. A nonparametric Bayesian model places a prior distribution over the infinite-dimensional parameter space. This allows the dimensionality of the model parameters to adapt to the complexity of the data set, thus protecting against over-fitting while allowing enough parameters to prevent under-fitting. After observing the data, since the data are always finite, nonparametric Bayesian models fit the data with finite dimensional parameters.

Although Bayesian approaches, especially nonparametric Bayesian approaches, provide a principled way of parameter tuning and model selection, they do not address the bias issue of clustering. Bayesian approaches have a built-in model selection criterion, the marginal

likelihood. However, there's no ground-truth to validate this criterion.

An additional issue with clustering is high dimensionality. High-dimensional data, e.g., text data, pose a difficult challenge to the clustering process. Usually clustering algorithms can handle data with low dimensionality, but as the dimensionality of the data increases, most algorithms do not scale well. The higher dimensionality, the sparser the data.

A common scenario with high-dimensional data is that clusters may exist in different subspaces comprised of different combinations of features (dimensions). In other words, data points in a cluster may be similar to each other along a subset of dimensions, but not in all dimensions. At the same time, data points located in another cluster may form a tight group with respect to different subset of dimensions. Therefore, such clusters are not defined in terms of all dimensions, but subsets of dimensions. The resulting clusters are called *subspace clusters*. Common global dimensionality reduction techniques are unable to capture such local structure of the data. Thus, a proper feature selection procedure should operate locally in the high-dimensional feature space.

People have proposed *subspace clustering* techniques, a.k.a. *co-clustering* or *bi-clustering*, to address the dimensionality issue (here, I use the term co-clustering). Often data can be organized in a matrix, such data are called *dyadic data*, where rows and columns represent objects and features, or different objects, respectively. The entries in the matrix represent the binary relation between the corresponding rows and columns, i.e., values for features given each object, or strength of relations between each pair of objects. In dyadic data, each row is usually represented in terms of all columns, and vice versa. If the number of rows and columns is large, this representation is of high dimensionality. Co-clustering is to cluster rows and columns into row- and column-clusters, given dyadic data organized in a matrix. The resulting co-clusters, defined in terms of row- and column-clusters, group similar entries together, while entries in different co-clusters are dissimilar. The co-clustering result can be viewed as a partition over the dyadic data matrix. One can then represent each row in terms of column-clusters, and each column in terms of row-clusters. Since the number of row- and column-clusters is much less than the number of rows and columns, co-clustering

reduces the dimensionality of original matrix.

Like clustering, also co-clustering suffers from the ill-posed nature and the lack of ground-truth to validate the results. In fact, different co-clustering algorithms might use different similarity criteria, and therefore lead to various co-clustering results with different biases. Similarly, model selection and parameter tuning is still an issue for co-clustering.

In summary, the aforementioned three issues make clustering a challenging problem. Work has been done to address each issue individually. Cluster ensembles have been proposed to address the different bias problem. But clustering ensembles still suffer from the model selection and parameter tuning issue. Nonparametric Bayesian approaches have been applied to clustering to perform model selection and parameter tuning, but they do not address the bias issue. Further, subspace clustering, or co-clustering, has been proposed to address the dimensionality problem. But co-clustering still suffers from the ill-posed nature. Thus, although attempts have been made in the literature to address individually the major issues related to clustering, no previous work has addressed them jointly.

1.2 Problem Statement

The work conducted in this dissertation narrows the research gap outlined in the previous section. Specifically, I tackled the problem represented by the three long-standing issues of clustering, namely the different bias, the model selection and parameter tuning, and the high dimensionality.

To this end, I introduced a new unified framework that addresses all three issues discussed above at the same time. This is a non-trivial task as it involves solving a new problem altogether: the co-clustering ensemble problem. The proposed framework combines and leverages the ensemble methodology, co-clustering and nonparametric Bayesian learning techniques.

1.3 Contributions

I designed a nonparametric Bayesian clustering ensemble (NBCE) approach, which assumes that multiple observed clustering results are generated from an unknown consensus clustering. The underlying distribution is assumed to be a mixture distribution with a nonparametric Bayesian prior, i.e., a Dirichlet Process. The number of mixture components, a.k.a. the number of consensus clusters, is learned automatically. By combining the ensemble methodology and nonparametric Bayesian modeling, NBCE addresses both the ill-posed nature and the parameter setting/model selection issues of clustering. Furthermore, NBCE outperforms individual clustering methods, since it can escape local optima by combining multiple clustering results.

I also designed a nonparametric Bayesian co-clustering ensemble (NBCCE) technique. NBCCE inherits the advantages of NBCE, and in addition it is effective with high dimensional data. As such, NBCCE provides a unified framework to address all the three aforementioned issues. NBCCE assumes that multiple observed co-clustering results are generated from an unknown consensus co-clustering. The underlying distribution is assumed to be a mixture with a nonparametric Bayesian prior. I developed two models to generate co-clusters in terms of row- and column- clusters. In one case row- and column-clusters are assumed to be independent, and NBCCE assumes two independent Dirichlet Process priors on the hidden consensus co-clustering, one for rows and one for columns. The second model captures the dependence between row- and column-clusters by assuming a Mondrian Process prior on the hidden consensus co-clustering. Combined with Mondrian priors, NBCCE provides more flexibility to fit the data.

In addition, existing co-clustering techniques, including NBCCE, typically only leverage the entries of the given contingency matrix to perform the two-way clustering. As a consequence, they cannot predict the interaction values for new objects. Predictions can only be made for objects already observed. In many applications, additional features associated to the objects of interest are available, e.g., sequence information for proteins. Such features can be leveraged to perform predictions on new data. Infinite Hidden Relational Model (IHRM)

[86] has been proposed to make use of features associated to the rows and columns of the contingency matrix. IHRM has the fundamental capability of forecasting relationships among previously unseen data. However, the original work of IHRM [86] didn't demonstrate how effective object features are in predicting relationships of unseen data. Here, I re-interpret IHRM from a co-clustering point of view, and demonstrate the ability of features to predict relationships of unseen objects on protein-molecule interaction data.

The contribution of my dissertation can be briefly summarized as follows:

- *Nonparametric Bayesian Clustering Ensembles*: I propose a nonparametric clustering ensemble approach based on a Dirichlet Processes Mixture model;

- *Nonparametric Bayesian Co-clustering Ensembles*: I propose two nonparametric Bayesian co-clustering ensemble approaches, one based on two independent Dirichlet Processes, the other based on Mondrian Processes;

- *Feature Enriched Dirichlet Process Co-clustering*: I evaluate the performance of Dirichlet Process Co-clustering when enriched with object features to measure the improvement achieved in predicting relationships between unseen objects.

The remaining chapters of this dissertation are organized as follows: Chapter 2 introduces the background and Chapter 3 discusses the related work. Chapter 4 and 5 introduce my work on nonparametric Bayesian clustering ensembles and co-clustering ensembles. Chapter 6 introduces the feature enriched Dirichlet Process co-clustering model. Chapter 7 summarizes the dissertation and discusses some future work.

Chapter 2: Background

In this chapter, I briefly introduce the background of my dissertation. First I'll introduce the problems I'll focus on, clustering, clustering ensemble and co-clustering. Then I'll introduce the methods I'm using, nonparametric Bayesian approaches, to solve those problems.

2.1 Clustering, Clustering Ensembles and Co-clustering

Clustering is to find a cluster structure for unlabeled data. A cluster is usually a collection of "similar" data while data belonging to different clusters are considered "dissimilar". So clustering tries to group similar data together, and dissimilar data into different clusters. Here, a natural question arises: what's a good clustering? Unfortunately, there is no absolute "best" criterion. As for text clustering, one could cluster documents by content, or by style. When the clustering criterion differs, similarity might differ, further clustering result will be different.

Clustering algorithms fall into three distinct types [27]: combinatorial algorithms, mode seeking, and mixture modeling. Combinatorial algorithms find (local) optimal clusterings by solving combinatorial optimization problems, without probabilistic modeling. Mode seeking attempts to find distinct modes of the probability density function from which observations are assumed to generate, then observations near to each mode comprise each cluster. Mixture modeling supposes that data are *i.i.d.* samples drawn from some mixture of components. Each component is defined by a parameterized probability density function; observations generated from the same density are considered within the same cluster. Therefore, mixture modeling converts clustering problems into density estimation problems. In this dissertation, I focus on mixture modeling for clustering via nonparametric Bayesian approaches.

Clustering ensembles [71] is proposed to address the different bias issue of clustering.

Clustering ensembles combine multiple base clusterings of a set of data into a single consensus clustering without accessing the features of data. By combining multiple clustering results, the combination process allows to cancel out emergent spurious structures that arise due to the various biases to which each individual clustering is tuned, or to the variance induced by different local optima. Therefore, clustering ensembles can provide solutions that are robust against spurious elements in the data.

A clustering ensemble technique is characterized by two components: the algorithm to generate base clusterings, and the machinery to combine the input clusterings into a consensus clustering. Base clustering results are typically generated by using different clustering algorithms [1], or by applying a single algorithm with various parameter settings [18, 39, 40], possibly in combination with data or feature sampling [16, 52, 78, 79].

There are two ways of modeling clustering ensemble problems. One is formalized as a combinatorial optimization problem, as in [71]; the other is formalized by mixture modeling, as in [84]. I focus on mixture modeling for clustering ensembles via nonparametric Bayesian approaches.

Often, the data themselves can manifest various structures, which may be hard to capture using a traditional clustering approaches. Consider dyadic data, e.g., documents and words, which can be represented by a matrix, whose rows correspond to documents and the columns correspond to words, and an entry is the term frequency of a word that appears in a document. If one wants to cluster both documents and words, one possible way may be to cluster the rows and columns independently using traditional clustering approaches. However, such simple way might fail to discover subtle patterns of the data, e.g., some sets of words may only appear in certain sets of documents, which means the data matrix may depict a block structure. In order to deal with such kind of structures, researchers proposed *co-clustering* algorithms [10, 12, 26, 49, 70, 85], which aim at taking into account information about columns while clustering rows, and vise versa. Given an $m \times n$ data matrix, co-clustering algorithms find co-clusters, where each co-cluster is a submatrix that manifest a similar pattern across a subset of rows and columns. Other nomenclature

for co-clustering include *biclustering*, *bidimensional clustering*, and *subspace clustering*.

Similarly to clustering approaches, co-clustering approaches fall into two distinct types: combinatorial algorithms [10, 12, 26] and mixture modeling [49, 70, 85]. Again, I focus on mixture modeling for co-clustering via nonparametric Bayesian approaches.

2.2 Bayesian Mixture Modeling

Mixture models have been extensively applied to clustering and classification. The basic mixture model for *i.i.d.* observations $\vec{X} = \langle x_i | i \in \{1, \cdots, N\} \rangle$ have the following probability density function:

$$p(x_i | K, \vec{w}, \vec{\theta}) = \sum_{k=1}^{K} w_k f(x_i | \theta_k) \tag{2.1}$$

where $f(\cdot|\theta)$ is a given parametric family of densities indexed by a scalar or vector parameter θ, such as the Gaussian, the Gamma, or the Poisson family; K is the unknown number of components; w_k is the component weight. The component weights are non-negative real numbers, subject to $\sum_{k=1}^{K} w_k = 1$. Let $\vec{w} = \langle w_1, \cdots, w_K \rangle$ and $\vec{\theta} = \langle \theta_1, \cdots, \theta_K \rangle$.

Mixture models represent a population consisting of subpopulations $k = 1, \cdots, K$ with sizes proportional to w_k. Random sampling from the population amounts to randomly choosing a subpopulation with probability proportional to its weight, and then drawing an observations from the subpopulation density. However, the identity of the subpopulation from which each observation is drawn is unknown. Therefore, it is natural to consider the group indicator z_i for the i-th observation as a latent variable. For $\vec{Z} = \langle z_1, \cdots, z_N \rangle$, the probability of $z_i = k$ is:

$$p(z_i = k) = w_k \tag{2.2}$$

Given the values of the z_i, the observations are drawn from their respective individual

subpopulations:

$$x_i|\vec{Z} \sim f(\cdot|\theta_{z_i}) \tag{2.3}$$

The formulation given by Equations (2.2) and (2.3) is convenient for calculation and interpretation. Integrating z out from Equations (2.2) and (2.3) brings back to Equation (2.1).

This representation in terms of latent indicators is called *completing the sample*. Following the EM terminology, $\vec{Z}$ and $\vec{X}$ are referred as the complete data. A natural model for clustering is to assume data are generated from such a mixture model. Here data generated from the same component are considered as a cluster; then $\vec{Z}$ becomes the vector of cluster assignments for the observed data.

In a Bayesian framework, the unknown K, $\vec{w}$ and $\vec{\theta}$ are regarded as drawn from appropriate prior distributions, denoted by $p(K, \vec{w}, \vec{\theta})$. The prior is assumed to be exchangeable for each component k, that is, invariant under permutations of the pairs (w_k, θ_k).

The likelihood function for the mixture model, denoted as $L(\vec{w}, \vec{\theta})$, is:

$$L(K, \vec{w}, \vec{\theta}) = \prod_{i}^{N} p(x_i|K, \vec{w}, \vec{\theta}) \tag{2.4}$$

The posterior density, which is our starting point for inference, is thus proportional to $L(K, \vec{w}, \vec{\theta})p(K, \vec{w}, \vec{\theta})$. Realistic models typically also involve hyperparameters. If I put distributions on hyperparameters, it does complicate inference.

In a Bayesian framework, mixture model inference, in essence, is mixture density estimation, where one needs to inference the number of components K, the component weights $\vec{w}$ and the density of each component $f(\cdot|\theta_k)$ (namely the parameter θ of each component).

2.3 Nonparametric Bayesian Models

2.3.1 Dirichlet Processes

The Dirichlet process (DP) [14] is an infinite-dimensional generalization of the Dirichlet distribution. Formally, let S be a set, G_0 a measure on S, and α_0 a positive real number. The random probability distribution G on S is distributed as a DP with concentration parameter α_0 (also called the pseudo-count) and base measure G_0 if, for any finite partition $\{B_k\}_{1\leq k\leq K}$ of S:

$$\begin{aligned}(G(B_1), G(B_2), \cdots, G(B_K)) \sim \\ \text{Dir}(\alpha_0 G_0(B_1), \alpha_0 G_0(B_2), \cdots, \alpha_0 G_0(B_K))\end{aligned}$$

Let G be a sample drawn from a DP. Then with probability 1, G is a discrete distribution [14]. Further, if the first $N-1$ draws from G yield K distinct values $\theta^*_{1:K}$ with multiplicities $n_{1:K}$, then the probability of the N^{th} draw conditioned on the previous $N-1$ draws is given by the Pólya urn scheme [5]:

$$\theta_N = \begin{cases} \theta^*_k, & \text{with prob } \frac{n_k}{N-1+\alpha_0},\ k \in \{1, \cdots, K\} \\ \theta^*_{K+1} \sim G_0, & \text{with prob } \frac{\alpha_0}{N-1+\alpha_0} \end{cases}$$

The DP is often used as a nonparametric prior in Bayesian mixture models [2]. Assume the data are generated from the following generative process:

$$\begin{aligned} G &\sim \text{Dir}(\alpha_0, G_0) \\ \theta_{1:N} &\sim G \\ x_{1:N} &\sim \prod_{n=1}^{N} F(\cdot|\theta_n), \end{aligned}$$

where the $F(\cdot|\theta_n)$ are probability distributions known as mixture components. Typically, there are duplicates among the $\theta_{1:N}$; thus, multiple data points are generated from the same mixture component. It is natural to define a cluster as those observations generated from a given mixture component. This model is known as the *Dirichlet process mixture* (DPM) model. Although any finite sample contains only finitely many clusters, there is no bound on the number of clusters and any new data point has non-zero probability of being drawn from a new cluster [54]. Therefore, DPM is known as an "infinite" mixture model.

The DP can be generated via the stick-breaking construction [68]. Stick-breaking draws two infinite sequences of independent random variables, $v_k \sim \text{Beta}(1, \alpha_0)$ and $\theta_k^* \sim G_0$ for $k = \{1, 2, \cdots\}$. Let G be defined as:

$$\pi_k = v_k \prod_{j=1}^{k-1} (1 - v_j) \tag{2.5}$$

$$G = \sum_{k=1}^{\infty} \pi_k \delta(\theta_k^*) \tag{2.6}$$

where $\vec{\pi} = \langle \pi_k | k = 1, 2, \cdots \rangle$ are mixing proportions and $\delta(\theta)$ is the distribution that samples the value θ with probability 1. Then $G \sim \text{Dir}(\alpha_0, G_0)$. It is helpful to use an indicator variable z_n to denote which mixture component is associated with x_n. The generative process for the DPM model using the stick-breaking construction is:

1. Draw $v_k \sim \text{Beta}(1, \alpha_0)$, $k = \{1, 2, \cdots\}$ and calculate $\vec{\pi}$ as in Eq (2.5).
2. Draw $\theta_k^* \sim G_0$, $k = \{1, 2, \cdots\}$
3. For each data point $n = \{1, 2, \cdots, N\}$:
   - Draw $z_n \sim \text{Discrete}(\vec{\pi})$
   - Draw $x_n \sim F(\cdot|\theta_{z_n}^*)$

The most popular inference method for DPM is MCMC [54]. Here we briefly introduce

Gibbs sampling for DPM when $F(\cdot|\theta^*_{z_n})$ and G_0 are conjugate. Conditioned on observations $\{x_n\}_{n\in\{1,\cdots,N\}}$ sampled from G and values $\{z_n\}_{n\in\{1,\cdots,N\}}$ for the indicator variables, the posterior density function for the parameter θ^*_k for the k^{th} cluster is also a member of the conjugate family:

$$p(\theta^*_k|\{x_n, z_n\}_{n\in\{1,\cdots,N\}}) = g(\theta^*_k|\zeta^*_k) = \tag{2.7}$$

$$\frac{\prod_{n=1}^{N} f(x_n|\theta^*_k)^{1_{[z_n=k]}} g(\theta^*_k|\zeta_0)}{\int \prod_{n=1}^{N} f(x_n|\theta^*_k)^{1_{[z_n=k]}} g(\theta^*_k|\zeta_0) d\theta^*_k}$$

where $1_{[\cdot]}$ is the indicator function, $f(x|\theta)$ is the density (or mass) function for $F(\cdot|\theta)$, $g(\theta|\zeta_0)$ is the density function for G_0, and $g(\theta^*_k|\zeta^*_k)$ is the posterior density function, with hyperparameter ζ^*_k obtained using the conjugate updating rule. Conditioned on the next indicator variable z_{N+1}, the predictive distribution for the next data point is given by:

$$p(x_{N+1}|\{x_n, z_n\}_{n\in\{1,\cdots,N\}}, z_{N+1} = k) \tag{2.8}$$

$$\int f(x_{N+1}|\theta^*_k) g(\theta^*_k|\zeta^*_k) d\theta^*_k,$$

can also be obtained in closed form. Having integrated out the parameters, it is necessary to Gibbs sample only the indicator variables. The conditional probability for sampling the indicator variable for the i^{th} data point is given as follows. For populated clusters $k \in \{z_n\}_{n\in\{1,\cdots,i-1,i+1,\cdots,N\}}$,

$$p(z_i = k|x_i, \{x_n, z_n\}_{n\in\{1,\cdots,i-1,i+1,\cdots,N\}}) \tag{2.9}$$

$$\propto \frac{n_k^{\neg i}}{N - 1 + \alpha_0} \int f(x_i|\theta^*_k) g(\theta^*_k|\zeta^{*\neg i}_k) d\theta^*_k.$$

Here, $n_k^{\neg i}$ is the number of data points other than x_i assigned to the k^{th} cluster, and $g(\theta^*_k|\zeta^{*\neg i}_k)$

is the posterior density for the k^{th} cluster parameter given all observations except x_i. For unpopulated clusters $k \notin \{z_n\}_{n\in\{1,\cdots,i-1,i+1,\cdots,N\}}$, the predictive probability is:

$$p(z_i = k|x_i, \{x_n, z_n\}_{n\in\{1,\cdots,i-1,i+1,\cdots,N\}}) \tag{2.10}$$
$$\propto \frac{\alpha_0}{N-1+\alpha_0} \int f(x_i|\theta_k^*) g(\theta_k^*|\zeta_0) d\theta_k^*.$$

Eq (2.9) is the probability of assigning x_i to the k^{th} existing cluster, while Eq (2.10) is the probability of assigning x_i to its own singleton cluster. Additional details on DPM inference can be found in [54, 59].

2.3.2 Mondrian Processes

A Mondrian process $\mathcal{M} \sim MP(\lambda, (a, A), (b, B))$ on a 2-dimensional rectangle $(a, A) \times (b, B)$ generates random partitions of a rectangle as follows [65]. The parameter λ, called the *budget*, controls the overall number of cuts in the partition. At each stage, a random cost E is drawn and compared to the budget. If E exceeds the budget, the process halts with no cuts; otherwise, a cut is made at random, the cost is subtracted from the budget, and the process recurses on the two sub-rectangles, each being drawn independently from its own MP distribution.

The cost E of cutting the rectangle $(a, A) \times (b, B)$ is distributed exponentially with mean equal to $1/(A - a + B - b)$, the inverse of the combined length of the sides. That is, for fixed λ, a longer perimeter tends to result in a loIr cost. The parameter λ can be vieId as a rate of cut generation per unit length of perimeter. If a cut is made, it has horizontal or vertical direction with probability proportional to the lengths of the respective sides, and its placement is uniformly distributed along the chosen side. After a cut is made, a new budget $\lambda' = \lambda - E$ is calculated, and the sub-rectangles are independently partitioned according to a Mondrian process with rate λ'. That is, if the cut splits the horizontal side into (a, x) and (x, A), then the two sub-rectangle processes are $\mathcal{M}_< \sim MP(\lambda', (a, x), (b, B))$ and $\mathcal{M}_> \sim$

Algorithm 1 Mondrian $\mathcal{M} \sim MP(\lambda, (a, A), (b, B))$

let $\lambda' \leftarrow \lambda - E$ where $E \sim \text{Exp}(A - a + B - b)$
if $\lambda' < 0$ **then**
  return $\mathcal{M} \leftarrow \{(a, A) \times (b, B)\}$
end if
draw $\rho \sim \text{Bernoulli}(\frac{A-a}{A-a+B-b})$
if $\rho = 1$ **then**
  draw $x \sim \text{Uniform}(a, A)$
  let $\mathcal{M}_1 \leftarrow MP(\lambda', (a, x), (b, B))$
  let $\mathcal{M}_2 \leftarrow MP(\lambda', (x, A), (b, B))$
  return $\mathcal{M} \leftarrow \mathcal{M}_1 \cup \mathcal{M}_2$
else
  draw $x \sim \text{Uniform}(b, B)$
  let $\mathcal{M}_1 \leftarrow MP(\lambda', (a, A), (b, x))$
  let $\mathcal{M}_2 \leftarrow MP(\lambda', (a, A), (x, B))$
  return $\mathcal{M} \leftarrow \mathcal{M}_1 \cup \mathcal{M}_2$
end if

$MP(\lambda', (x, A), (b, B))$, respectively. Conversely, for a vertical cut into (b, x) and (x, B), the sub-rectangle processes are $\mathcal{M}_< \sim MP(\lambda', (a, A), (b, x))$ and $\mathcal{M}_> \sim MP(\lambda', (a, A), (x, B))$.

The one-dimensional Mondrian process reduces to a Poisson process. The MP shares with the Poisson process the self-consistency property that its restriction to a subspace is a Mondrian process with the same rate parameter as the original Mondrian process. As with the Poisson process, one can define a non-homogeneous MP by sampling the cuts non-uniformly according to a measure defined along the sides of the rectangle [65]. This work considers only the homogeneous MP.

Algorithm 1 describes how to sample from the MP with rate λ on a 2-dimensional space $(a, A) \times (b, B)$. More details on the Mondrian Process can be found in [65].

Relations and Exchangeability

Consider a stochastic 2-dimensional matrix $\vec{R} = \langle r_{i_1 \cdots i_n} \rangle$, where i and j index objects $x_i \in X$ and $y_j \in Y$ in possibly distinct sets X and Y. A binary matrix represens a relation on $X \times Y$, where $r_{i,j}$=1 ($r_{i,j}$=0) indicates presence(absence) of a relationship between x_i and y_j. More generally, if $r_{i,j} \in \mathcal{R}$, the matrix $\vec{R}$ represents a function from $X \times Y$ to $\mathcal{R}$.

The stochastic matrix $\vec{R}$ is *separately exchangeable* if its distribution is invariant to

separate permutations of rows and columns [65]. That is, let $\vec{R}_{1:n,1:m}$ be an n by m matrix; let $\pi(1:n)$ and $\sigma(1:m)$ denote permutations of the integers from 1 to n and 1 to m respectively; and let $\vec{R}_{\pi(1:n),\sigma(1:m)}$ denote the matrix obtained from $\vec{R}_{1:n,1:m}$ by permuting its rows according to $\pi(1:n)$ and its columns according to $\sigma(1:m)$. Separate exchangeability means that for any permutation $\pi(1:n)$ and $\sigma(1:m)$, the distribution of $\vec{R}_{1:n,1:m}$ is the same as the distribution of $\vec{R}_{\pi(1:n),\sigma(1:m)}$. That is, the specific association of objects to indices is irrelevant to the distribution.

The distribution of a separately exchangeable relation can be parameterized in terms of a latent parameter for each dimension and an additional random parameter. This representation, originally due to Aldous and Hoover, was exploited by [65] to model relational data using the Mondrian process. In the 2-dimensional case, I have a latent parameter ξ_i for each $x_i \in X$, a latent parameter η_j for each $y_j \in Y$, and an additional random parameter θ. The ξ_i are iid draws from p_ξ; the η_j are iid draws from p_η; and θ is drawn from p_θ. Then [65],

$$p(\vec{R}_{1:n,1:m}) = \int p_\theta(\theta) \prod_i p_\xi(\xi_j) \times \prod_j p_\eta(\eta_j) \prod_{i,j} p_{\vec{R}}(r_{i,j}|\theta, \xi_i, \eta_j) d\theta d\xi_{1:n} d\eta_{1:m} \tag{2.11}$$

2.4 Advanced Bayesian Inference

2.4.1 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) [53] is a very popular inference method for Bayesian models. MCMC first builds a Markov chain whose stationary distribution is the target posterior distribution, and then draws samples from the Markov chain. The law of large numbers justifies the use of sample averages as estimations.

A Markov chain is described by a transition kernel $P(\theta, d\theta')$ that specifies for each state

θ the probability of jumping to state θ' in next step, where $\theta, \theta' \in \Theta$, and Θ the state space. Here, one assumes that for each transition distribution the corresponding density function exists and denote the transition density as $p(\theta, \theta')$.

MCMC requires that the constructed chain must be aperiodic, irreducible and reversible. Especially, the reversibility condition, known as "detailed balance" equation, is:

$$\pi(\theta)p(\theta, \theta') = \pi(\theta')p(\theta', \theta) \tag{2.12}$$

where π is the initial distribution of the starting state. One can derive that:

$$\int \pi(\theta)p(\theta, \theta')d\theta = \pi(\theta') \tag{2.13}$$

which means that π is a stationary distribution of the produced chain. Further, aperiodicity and irreducibility ensure that π is the unique stationary distribution. Therefore, a sample drawn from the chain is considered as a sample drawn from the distribution π. For Bayesian inference, the target distribution π is set to be the posterior distribution of interest.

Metropolis-Hastings Sampling

Metropolis-Hastings (MH) sampling [28, 50] is the most important MCMC approach. Any variant of MCMC methods can be considered as a special case of MH. MH constructs a Markov chain with transition density:

$$p(\theta, \theta') = q(\theta, \theta')\alpha(\theta, \theta'), \text{ for } \theta \neq \theta' \tag{2.14}$$

$$p(\theta, \theta) = 1 - \int q(\theta, \theta')\alpha(\theta, \theta')d\theta \tag{2.15}$$

where q is the proposal density to propose a candidate jump-to state θ', and $\alpha(\theta, \theta')$ is the acceptance probability of accepting the jump from θ to θ'. According to the detailed balance

condition, the chain has to satisfy that:

$$\pi(\theta)q(\theta,\theta')\alpha(\theta,\theta') = \pi(\theta')q(\theta',\theta)\alpha(\theta',\theta) \tag{2.16}$$

Assuming $\alpha(\theta',\theta) = 1$, which leads to choose $\alpha(\theta,\theta')$ as

$$\alpha(\theta,\theta') = \min\left(1, \frac{\pi(\theta')q(\theta',\theta)}{\pi(\theta)q(\theta,\theta')}\right) \tag{2.17}$$

One can see that MH can only accept θ' with positive probability jumping back to θ.

MH samplers begin at an arbitrary state θ_1 and proceeds through a sequence of states $\theta_2, \theta_3, \cdots$, as follows. Conditioned the current state θ_i, a new state θ' is proposed according to the proposal distribution $q(\theta_i,\theta')$. The new state is accepted with probability computed by Eq. (2.17). If accepted, set the next state $\theta_{i+1} = \theta'$, otherwise, $\theta_{i+1} = \theta_i$.

Reversible Jump MCMC

For detailed balance to hold, both the forward and backward transition densities must be positive for any allowable transition. This is not the case when standard MH sampling is used to sample from different parameter spaces, such as parameter spaces of varying dimensionality. Reversible jump Markov Chain Monte Carlo (RJMCMC) [22, 23, 81] generalizes standard MH to problems in which the proposal and target distributions have densities on spaces of different dimensions.

First let's consider a unusual case, that the parameter spaces changed but the dimensionality of the spaces remains the same. For example, $\theta_i \in (0,1)$, while $\theta_{i+1} \in (-\infty,+\infty)$. Since the parameter spaces changed, a probability measure for θ_i is no longer a probability measure for θ_{i+1}, since they don't even have the same support. But if one can build a bijection between $(0,1)$ and $(-\infty,+\infty)$, e.g. $\theta_{i+1} = -\log(\frac{1}{\theta_i} - 1)$, the inverse of sigmoid function, then a probability measure for θ_i is also a probability measure for θ_{i+1}. Further, with the bijection, θ_i and θ_{i+1} can have the same probability measure, then standard MH

sampling can be applied, even θ_i and θ_{i+1} are of different space.

Next, let's consider trans-dimensional case, that the dimensionality of parameter spaces changes. Assume a state $s = (k, \vec{\theta}_k)$ consisting of a discrete subspace indicator k and a continuous, subspace-dependent parameter $\vec{\theta}_k$ whose dimensionality n_k depends on k. To ensure reversibility of moves between subspaces of different dimensions, auxiliary parameters are introduced to match dimensions of the current and proposed states. Specifically, associated with each pair k and k' of states, there are two auxiliary parameters $\vec{u}_{kk'}$ and $\vec{u}_{k'k}$ satisfying the dimension matching condition

$$n_k + n_{\vec{u}_{kk'}} = n_{k'} + n_{\vec{u}_{k'k}}. \tag{2.18}$$

Then, a bijective mapping $h_{kk'}$ is defined to transform the augmented state of subspace k to the augmented state of subspace k'. Accordingly, the inverse transform $h_{k'k}^{-1}$ maps from augmented states of subspace k' to augmented states of subspace k. Again, the bijection ensures the same probability measure for the the two augmented subspaces. RJMCMC draws samples on the augmented state space, thus always proposing moves between spaces of equal dimension.

RJMCMC for trans-dimensional case is described in Algorithm 2. A move from a state $(k, \vec{\theta}_k)$ in subspace k to a state $(k', \vec{\theta}_{k'})$ in subspace k' is proposed with probability $q(k, k')$. Conditioned on the move from k to k', two new augmented parameter vectors are proposed as follows. First, one new random auxiliary parameter $\vec{u'}_{kk'}$ is proposed from the density $q_{kk'}(\vec{u}_{kk'}|\vec{\theta}_k)$. Then, the other new random auxiliary parameter $\vec{u}_{k'k}$ is proposed from the density $q_{k'k}(\vec{u}_{k'k}|\vec{\theta}_{k'})$. Finally, the new state is accepted with probability

$$\alpha((k', \vec{\theta}_{k'}, \vec{u}_{k'k}), (k, \vec{\theta}_k, \vec{u}_{kk'})) = \tag{2.19}$$

$$\min\left(1, \frac{\pi(k', \vec{\theta}_{k'}) q(k', k) q_{k'k}(\vec{u}_{k'k}|\vec{\theta}_{k'})}{\pi(k, \vec{\theta}_k) q(k, k') q_{kk'}(\vec{u}_{kk'}|\vec{\theta}_k)} J\right),$$

where $\pi(k', \vec{\theta}_{k'})$ is the target distribution and J is the Jacobin calculated as:

$$J = \left| \frac{\partial h_{kk'}(\vec{\theta}_k, \vec{u}_{kk'})}{\partial \vec{\theta}_k \partial \vec{u}_{kk'}} \right|. \tag{2.20}$$

The Jacobin (2.19) is needed to ensure detailed balance. It's due to the change of variables of the probability density functions from one variable space to another variable space. Since here involve two augmented parameter space, although the bijection ensures the same probability measure for the two spaces, when evaluating the density function, it's required to evaluate the density in one parameter space, so here it changes the parameter in the augmented space of k to the augmented space of k', and the variable change results in the Jacobin. In many cases of interest, the bijection is just the identity mapping, and J is just 1.

Algorithm 2 Reversible Jump MCMC

Initialize model k.
repeat
  Propose a new model k' by drawing it from the distribution $p(k, \cdot)$,
  Propose the parameter for the new model by generating $\vec{u'}_{kk'}$ from distribution $q_{kk'}(\cdot|\vec{\theta}_k)$, and set $(\vec{\theta}_{k'}, \vec{u'}_{k'k}) = h_{kk'}(\vec{\theta}_k, \vec{u'}_{kk'})$,
  Randomly choose whether to accept the new state according to the acceptance probability given in (2.19). If move is accepted, set $k = k'$.
until Stopping criterion is met.

Gibbs sampling

Often, model parameter space is of high dimensionality. Assume model parameters are n-dimensional vectors, denoted as $\vec{\theta} = \langle \theta_1, \cdots, \theta_n \rangle$. One way of proposing a new parameter $\vec{\theta'}$ conditioned on current parameter $\vec{\theta}$, is just change one dimension of $\vec{\theta}$, which means $\vec{\theta'}$ is the same as $\vec{\theta}$ except on one of the dimensions. Without loss of generality, assume $\vec{\theta'}$ is different as $\vec{\theta}$ on the i^{th} dimension, that $\vec{\theta'} = \langle \theta_1, \cdots, \theta'_i, \cdots, \theta_n \rangle$. The proposal distribution $q(\vec{\theta}, \vec{\theta})$ only needs to proposal a new value on the i^{th} dimension. Denote the proposal as

$q(\theta_i', \vec{\theta})$, and the accept ratio changes to:

$$\alpha(\vec{\theta}, \vec{\theta'}) = \min\left(1, \frac{\pi(\vec{\theta'})q(\theta_i', \vec{\theta})}{\pi(\vec{\theta})q(\theta_i, \vec{\theta'})}\right) \tag{2.21}$$

It's turned out that if the proposal $q(\theta_i', \vec{\theta})$ is the conditional distribution of θ_i conditioned on all other dimensions, the accept ratio becomes 1. The conditional distribution of θ_i conditioned on all other dimensions is:

$$p(\theta_i|\theta_1, \cdots, \theta_{i-1}, \theta_{i+1}, \cdots, \theta_n) = \frac{\pi(\vec{\theta})}{\pi(\vec{\theta}^{\neg i})} = \frac{\pi(\theta_1, \cdots, \theta_n)}{\pi(\theta_1, \cdots, \theta_{i-1}, \theta_{i+1}, \cdots, \theta_n)} \tag{2.22}$$

where $\vec{\theta}^{\neg i}$ denotes $\vec{\theta}$ excludes the i^{th} dimension.

Then $\pi(\vec{\theta'})q(\theta_i, \vec{\theta'})$ becomes:

$$\pi(\vec{\theta'})q(\theta_i, \vec{\theta'}) = \tag{2.23}$$
$$\pi(\theta_1, \cdots, \theta_i', \cdots, \theta_n)p(\theta_i|\theta_1, \cdots, \theta_{i-1}, \theta_{i+1}, \cdots, \theta_n) = \pi(\theta_1, \cdots, \theta_i, \theta_i', \cdots, \theta_n)$$

and $\pi(\vec{\theta})q(\theta_i', \vec{\theta})$ becomes:

$$\pi(\vec{\theta})q(\theta_i', \vec{\theta}) = \tag{2.24}$$
$$\pi(\theta_1, \cdots, \theta_i, \cdots, \theta_n)p(\theta_i'|\theta_1, \cdots, \theta_{i-1}, \theta_{i+1}, \cdots, \theta_n) = \pi(\theta_1, \cdots, \theta_i, \theta_i', \cdots, \theta_n)$$

Since $\pi(\vec{\theta'})q(\theta_i, \vec{\theta'}) = \pi(\vec{\theta})q(\theta_i', \vec{\theta})$, the accept ratio becomes 1.

This kind of sampling method is called *Gibbs sampling*, which accepts every proposed new state. So no need to explicitly calculate the accept ratio for Gibbs sampling, but directly draw samples from the conditional distribution $p(\theta_i|\theta_1, \cdots, \theta_{i-1}, \theta_{i+1}, \cdots, \theta_n)$.

In practice, Gibbs sampling updates each dimension of model parameters one by one,

which is to update the first dimension by drawing sample θ_1' from $p(\theta_1|\theta_2, \cdots, \theta_n)$, then draw sample θ_2' from $p(\theta_2|\theta_1', \theta_3, \cdots, \theta_n)$, note that the first dimension has been updated from θ_1 to θ_1', and then so on so forth, until update all dimensions.

2.4.2 Variational Bayesian Inference

Variational Bayesian (VB) inference [4,35] approximates an intractable posterior distribution using a tractable distribution, called variational distribution. By doing so, VB converts an inference problem to an optimization problem, in that VB finds the best approximation to the posterior in a tractable solution space, not in an intractable one.

I assume observed data $\vec{X} = \langle x_1, \cdots, x_n \rangle$, corresponding latent variables $\vec{Z} = \langle z_1, \cdots, z_n \rangle$ and parameters $\vec{\theta}$ of a model m. Further, I assume a prior $p(\vec{\theta}|m)$ to the model parameters. VB finds a lower bound to the log-likelihood function of the model m which is:

$$\log p(\vec{X}|m) = \log \int p(\vec{X}, \vec{Z}, \vec{\theta}|m) d\vec{\theta} d\vec{Z} = \log \int q(\vec{Z}, \vec{\theta}) \frac{p(\vec{X}, \vec{Z}, \vec{\theta}|m)}{q(\vec{Z}, \vec{\theta})} d\vec{\theta} d\vec{Z} \quad (2.25)$$

$$\geq \int q(\vec{Z}, \vec{\theta}) \log \frac{p(\vec{X}, \vec{Z}, \vec{\theta}|m)}{q(\vec{Z}, \vec{\theta})} d\vec{\theta} d\vec{Z} \quad (2.26)$$

where $q(\vec{Z}, \vec{\theta})$ is the variational distribution. For the sake of tractability, $\vec{Z}$ and $\vec{\theta}$ are assumed to be independent in the variational distribution, and then $q(\vec{Z}, \vec{\theta}) = q(\vec{Z})q(\vec{\theta})$. Further, assume every z_i of $\vec{Z}$ is independent, that $q(\vec{Z}) = \prod_{i=1}^{n} q(z_i)$. I can rewrite Equation (2.26) as:

$$\log p(\vec{X}|m) \geq \int q(\vec{\theta}) \left(\int q(\vec{Z}) \log \frac{p(\vec{X}, \vec{Z}|\vec{\theta}, m)}{q(\vec{Z})} d\vec{Z} + \log \frac{p(\vec{\theta}|m)}{q(\vec{\theta})} \right) d\vec{\theta} \quad (2.27)$$

By taking derivative of Equation (2.27) w.r.t. $q(\vec{Z})$ and $q(\vec{\theta})$, respectively, and setting

the derivatives as zero, I can compute $q(\vec{Z})$ and $q(\vec{\theta})$ as following:

$$q(z_i) \propto \exp\left(\int q(\vec{\theta}) \log p(x_i, z_i|\vec{\theta}, m) d\vec{\theta}\right) \tag{2.28}$$

$$q(\vec{\theta}) \propto p(\vec{\theta}|m) \exp\left(\int q(\vec{X}) \log p(\vec{X}, \vec{Z}|\vec{\theta}, m) d\vec{Z}\right) \tag{2.29}$$

Collapsed variational Bayesian (CVB) inference [42, 74, 75] improves standard VB, by marginalizing out $\vec{\theta}$ before applying VB inference. By doing so, the variational distribution of CVB, $q(\vec{Z})$, involves only $\vec{Z}$, no $\vec{\theta}$. So CVB does not need to assume $\vec{Z}$ and $\vec{\theta}$ are independent in the variational distribution. Therefore, CVB has a less restricted assumption than VB, which allows CVB to search in a less restricted tractable solution space to find the (local) optima. Thus, CVB can find in general a better approximation to the posterior distribution than VB.

Chapter 3: Related Work

3.1 Clustering

Extensive work has been done on clustering. Discriminative approaches to clustering include k-means [45, 47], k-medians [33], fuzz clustering [29], spectral clustering [80], and hierarchical clustering [34]. An example of a generative approach to clustering is mixture of Gaussians [11]. Here I review two Bayesian clustering approaches based on mixture models, Dirichlet Processes [14] and Latent Dirichlet Allocation [9], which are closely related to my work.

3.1.1 Dirichlet Processes

Dirichlet Processes (DP) [14] have a long history. Each sample drawn from a DP is an infinite discrete distribution. There are some intuitive representations for Dirichlet Processes, one is Pólya urn schemes [5], another famous constructive definition is stick-breaking construction [68]. There are many generalizations to DP, such as Hierarchical DP [73], which assumes a DP as the base measure to several related DP's; Pitman-Yor Processes [60], whose stick-breaking representation differs from that of DP in that $v_i \sim Beta(a, b)$ instead of $v_i \sim Beta(1, \alpha_0)$; Nested DPs [64], from which each drawn sample is again a DP, and etc.

The stick-breaking construction can not only be used to construct DP, but also many other discrete nonparametric Bayesian models, e.g. Pitman-Yor Processes [60] and Indian Buffet Processes [24]. In [31], a Gibbs sampler was proposed for stick-breaking priors.

Recently [65] proposed a multidimensional non-parametric prior process, called Mondrian Processes, for modeling relational data. The process is based on "multidimensional stick-breaking", where in two-dimensional case, it generates non-overlapping axis-aligned cuts in a unit matrix.

3.1.2 Probabilistic Topic Modeling

"Latent Dirichlet Allocation" (LDA) proposed by Blei *et al.* [9] applied Bayesian mixture models to model text data. LDA first defines each topic as a mixture distribution over words, and then each document is assumed to be a mixture of topics. Each topic can be thought as a soft clustering of words, where similar and related words are grouped together. Further, LDA assumes that given topics, documents and words are conditionally independent. By doing so, LDA represents each document in term of topics, instead of words, which greatly reduces the dimensionality of document representation, since the number of topics is considerably smaller than the number of words.

A standard variational Bayesian (VB) algorithm [9] is used to estimate the posterior distribution of model parameters given the model evidence. The standard VB simplifies the true intractable posterior to a tractable approximation, which transforms the inference problem into an optimization one consisting in finding a best tractable approximation to the true posterior. Griffiths *et al.* proposed a collapsed Gibbs sampling method to learn the posterior distribution of parameters for the LDA model [25]. Recently, Teh *et al.* proposed a collapsed variational Bayesian (CVB) algorithm to perform model inference for LDA [75], which borrows the idea from the collapsed Gibbs sampling that first integrate out model parameters then perform standard VB. Recently, LDA model has been extended to supervised learning scenario, [8, 51, 62].

A nonparametric Bayesian version of the Latent Dirichlet Allocation (LDA) mode is called Dirichlet Enhanced Latent Semantic Analysis (DELSA) model [87]. DELSA treats documents as being organized around latent topics drawn from a mixture distribution with parameters drawn in turn from a Dirichlet process. The posterior distribution of the topic mixture for a new document converges to a flexible mixture model in which both mixture weights and mixture parameters can be learned from the data. Thus, the posterior distribution is able to represent the distribution of topics more robustly. After learning, typically only a few components have non-negligible weights; thus the model is able to naturally output clusters of documents.

Later, Blei *et al.* proposed Hierarchical Topic Model and Chinese Restaurant Process [6, 7], another nonparametric Bayesian topic model which can learn topic hierarchy in a unsupervised way, and recently proposed variational inference [83] for this model. [44] proposed a DAG-style hierarchical topic model, called Pachinko allocation, relaxing the assumption of fixed group assignments, and a nonparametric version proposed [43].

3.2 Clustering Ensembles

Ensemble methods have been a major success story in machine learning and data mining, particularly in classification and regression problems. Recent work has also focused on clustering, where ensembles can yield robust consensus clusterings [15, 17, 39, 71, 77]. Clustering ensembles combine various base clustering results and compute a consensus clustering, which is intended to be more robust and accurate than each individual base clustering result. Since these methods require only the base clustering results and not the raw data themselves, clustering ensembles provide a convenient approach to privacy preservation and knowledge reuse [84]. Such desirable aspects have generated intense interest in cluster ensemble methods.

Various approaches have been proposed to address the clustering ensemble problem. Our focus is on statistically oriented approaches. One popular methodology to build a consensus function utilizes a co-association matrix [1,18,52,79]. Such matrix can be seen as a similarity matrix, and thus can be used with any clustering algorithm that operates directly on similarities [1, 79]. In alternative to the co-association matrix, voting procedures have been proposed to build consensus clustering in [13]. Gondek et al. [21] derived a consensus clustering based on the Information Bottleneck principle: the mutual information between the consensus clustering and the individual input clusterings is maximized directly, without requiring approximation.

A different popular mechanism for constructing a consensus clustering maps the problem onto a graph-based partitioning setting [3, 30, 71]. In particular, Strehl et al. [71] proposed

three graph-based approaches: Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm (HGPA), and Meta-Clustering Algorithm (MCLA). The methods use METIS (or HMETIS) [36] to perform graph partitioning. The authors in [61] developed soft versions of CSPA, HGPA, and MCLA which allow to combine soft partitionings of data.

Another class of clustering ensemble algorithms is based on probabilistic mixture models [77, 84]. Topchy et al. [77] proposed a mixture-membership model for clustering ensembles, which modeled the clustering ensemble as a finite mixture of multinomial distributions in the space of base clusterings. A consensus result is found as a solution to the corresponding maximum likelihood problem using the EM algorithm. Wang et al. [84] proposed Bayesian Cluster Ensembles (BCE), a model that applied a Bayesian approach to discovering clustering ensembles. BCE addresses the over-fitting issue to which the maximum likelihood method is prone [77]. The BCE model is applicable to some important variants of the basic clustering ensemble problem: it can be adapted to handle missing values in the base clusterings; it can handle the requirement that the base clusterings reside on a distributed collection of hosts; and it can deal with partitioned base clusterings in which different partitions reside in different locations. Other clustering ensemble algorithms, such as the cluster-based similarity partitioning algorithm (CSPA) [71], the hypergraph partitioning algorithm (HGPA) [71], or k-means based algorithms [41] can handle one or two of these cases; however, none except the Bayesian method can address them all. However, like most clustering ensemble methods, BCE has the disadvantage that the number of clusters in the consensus clustering must be specified *a priori*. A poor choice can lead to under- or over-fitting.

3.3 Co-clustering

Researchers have proposed several discriminative and generative co-clustering models. Dhillon *et al.* [12] introduced an information-theoretic co-clustering approach based on hard partitions. Shafiei *et al.* [69] proposed a soft-partition co-clustering method called "Latent

Dirichlet Co-clustering." This model, however, does not cluster rows and columns simultaneously. A Bayesian Co-clustering (BCC) model has been proposed in [70]. BCC maintains separate Dirichlet priors for row- and column-cluster probabilities. To generate an entry in the data matrix, the model first generates the row and column clusters for the entry from their respective Dirichlet-multinomial distributions. The entry is then generated from a distribution specific to the row- and column-cluster. Like the original Latent Dirichlet Allocation (LDA) [9] model, BCC assumes symmetric Dirichlet priors for the data distributions given the row- and column-clusters. Shan and Banerjee [70] proposed a variational Bayesian algorithm to perform inference with the BCC model. In [85], the authors proposed a variation of BCC, "Latent Dirichlet Bayesian Co-clustering" (LDCC), and developed a collapsed Gibbs sampling and a collapsed variational Bayesian algorithm to perform inference. All aforementioned co-clustering models are parametric ones, i.e., they need to have specified the number of row- and column-clusters.

A nonparametric Bayesian co-clustering (NBCC) approach has been proposed in [49]. NBCC assumes two independent nonparametric Bayesian priors on rows and columns, respectively. As such, NBCC does not require the number of row- and column-clusters to be specified *a priori*. NBCC assumes a Pitman-Yor Process [60] prior, which is a generalization of Dirichlet Processes. Pitman-Yor processes, unlike Dirichlet processes, favor uniform cluster sizes. Existing Bayesian co-clustering models, e.g., BCC [70], LDCC [85], and NBCC [49], can handle missing entries only for already observed rows and columns.

Other researchers also applied Dirichlet Processes to relational learning, such as [37, 86]. The infinite relational model (IRM) [37] is very similar to NBCC, except that IRM can model not only binary relations between two different kinds of objects, but also binary relations between the same kind of objects. The infinite hidden relational model (IHRM) [86] leverages the features associated with each objects to better predict missing relations between objects, but IHRM didn't demonstrate how effective object features are in predicting relationships between unseen objects.

Co-clustering techniques have also been applied to collaborative filtering [66]. Collaborative filtering recommends items to users by discovering similarities among users based on their past consumption records, and using the discovered similarities to predict which items will be attractive to a user. The user consumption records can be organized in a matrix. Co-clustering techniques for collaborative filtering include the nearest bi-clustering method [72], evolutionary co-clustering for online collaborative filtering [38] and information-theoretic co-clustering [20].

Chapter 4: Nonparametric Bayesian Clustering Ensembles

4.1 Introduction

This chapter introduces a nonparametric Bayesian clustering ensembles model called Dirichlet Process-based Clustering Ensembles (DPCE). DPCE adapts the Dirichlet Process Mixture (DPM) model proposed by [54] to the clustering ensembles problem. DPCE allows the number of consensus clusters to be discovered from the data, while inheriting the desirable properties of the Bayesian clustering ensembles model [84]. Similar to the mixture modeling approach [77] and the parametric Bayesian approach [84], DPCE treats the base clustering results for each object as a feature vector with discrete feature values, and learns a mixed-membership model from this feature representation. An empirical evaluation (see Section 4.3 below) demonstrates the versatility and superior stability and accuracy of DPCE.

4.2 Dirichlet Process-based Clustering Ensembles

Following [77] and [84], we assume the observations are the output of M base clustering algorithms, each generating a hard partition on the N data items to be clustered. Let J_m denote the number of clusters generated by the m^{th} clustering φ_m, $m \in \{1, \cdots, M\}$, and let $y_{nm} \in \{1, \cdots, J_m\}$ denote the cluster ID assigned to the n^{th} data item x_n by φ_m, $n \in \{1, \cdots, N\}$. The row $\vec{y}_n = \langle y_{nm} | m \in \{1, \cdots, M\} \rangle$ of the base clustering matrix $\vec{Y}$ gives a new feature vector representation for the n^{th} data item. Following common practice in the clustering ensemble literature, [71], DPCE models the output of M base co-clusterings. The original data matrix $\vec{X}$, assumed inaccessible, is not modeled. Table 4.1 shows an example of base clusterings of DPCE.

Table 4.1: Example of Base Clusterings for DPCE

	φ_1	φ_2	$\cdots$	φ_m	$\cdots$	φ_M
x_1	2	A	$\cdots$	y_{1m}	$\cdots$	b
x_2	1	B	$\cdots$	y_{2m}	$\cdots$	a
x_3	2	B	$\cdots$	y_{3m}	$\cdots$	c
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
x_n	y_{n1}	y_{n2}	$\cdots$	y_{nm}	$\cdots$	y_{nM}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
x_N	3	A	$\cdots$	y_{Nm}	$\cdots$	d

4.2.1 DPCE Generative Model

Figure 4.1 depicts the generative model for DPCE. The observations $\vec{Y}$ are generated from a Dirichlet Process mixture model, where α_0 is the concentration parameter and G_m is the base measure for the m^{th} base clustering. The Dirichlet process is sampled via the stick-breaking construction as described in Eqs (2.5) and (2.6). The consensus cluster indicator variables z_n are iid draws of an integer-valued distribution parameterized by $\vec{\pi}$. A sequence $\vec{\theta}^*_{km}$ of parameters is drawn, for each consensus cluster $1 \leq k < \infty$ and base clustering $m \leq M$. These are drawn independently, with $\vec{\theta}^*_{km}$ having distribution G_m, where G_m is a symmetric J_m-dimensional Dirichlet distribution with total pseudo-count β_m.[1] Conditional on the indicator variables and unique parameter vectors, the cluster IDs are drawn independently. The cluster ID y_{mn} output by the m^{th} base clustering for the n^{th} datum is drawn from a discrete distribution with parameter $\vec{\theta}^*_{km}$, where k is equal to z_n, the consensus cluster indicator for the n^{th} datum.

Formally, the generative process for DPCE is:

- Draw $v_k \sim \text{Beta}(1, \alpha_0)$, for $k = 1, 2, \cdots, \infty$.
- Set mixture weights for consensus clusters $\pi_k = v_k \prod_{t=1}^{k-1}(1 - v_t)$, for $k = 1, 2, \cdots, \infty$.

[1] As the number of clusters is provided as input to typical clustering algorithms, J_m is treated as deterministic and known.

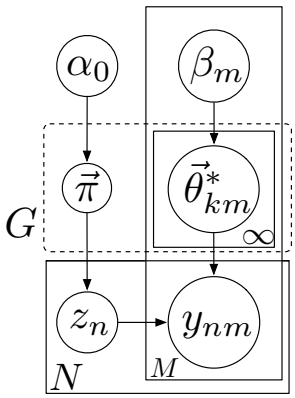


Figure 4.1: Dirichlet Process-based Clustering Ensemble Model

- For $k = 1, 2, \cdots, \infty$, $m = 1, 2, \cdots, M$, draw parameters for consensus clusters:

$$\vec{\theta}^*_{km} \sim \text{Dirichlet}(\frac{\beta_m}{J_m}, \cdots, \frac{\beta_m}{J_m})$$

- For each datum n:
  - Draw consensus cluster $z_n \sim \text{Discrete}(\vec{\pi})$;
  - For each base clustering φ_m, generate $y_{nm} \sim \text{Discrete}(\vec{\theta}^*_{z_n m})$.

4.2.2 DPCE Inference

We use the collapsed Gibbs sampling method discussed in Section 2.3.1 for DPCE inference. All model parameters except the concentration parameter α_0 are marginalized out. Only z_n and α_0 are sampled.

The conditional distribution for sampling z_n given $\vec{Y}$ and all other indicator variables $\vec{z}^{\neg n}$ is:

$$p(z_n = k|\vec{Y}, \vec{z}^{\neg n}, \alpha_0) \propto {\mathcal{N}_k}^{\neg n} \prod_{m=1}^{M} \frac{\mathcal{N}^{\neg n}_{k,y_{nm}} + \frac{\beta_m}{J_m}}{{\mathcal{N}_k}^{\neg n} + \beta_m} \tag{4.1}$$

when the cluster index k appears among the indices in $\vec{z}^{\neg n}$, and

$$p(z_n = k|\vec{Y}, \vec{z}^{\neg n}, \alpha_0) \propto \alpha_0 \prod_{m=1}^{M} \frac{1}{J_m} \tag{4.2}$$

when the cluster index k does not appear among the indices in $\vec{z}^{\neg n}$. Here, $\mathcal{N}_k^{\neg n}$ is the number of data points assigned to the k^{th} consensus cluster excluding the n^{th} datum, and $\mathcal{N}_{k,y_{nm}}^{\neg n}$ is the number of data points in the k^{th} consensus cluster that are also assigned to the same cluster as the n^{th} datum by φ_m, excluding the n^{th} datum.

To sample the concentration parameter α_0, we assign a Gamma prior to α_0, and perform Metropolis-Hastings sampling. Conditional on $\vec{z}$ and marginalizing over $\vec{\pi}$, α_0 is independent of the remaining random variables. As pointed out by [63], the number of observations N and the number of components K are sufficient for α_0. Following [2] and [73], we have:

$$p(K|\alpha_0, N) = s(N, K)\alpha_0^K \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + N)} \tag{4.3}$$

where $s(N, K)$ is the Stirling number. Treating (4.3) as the likelihood function for α_0, we assign a Gamma prior distribution $p(\alpha_0|a, b)$ for α_0. The posterior distribution for α_0 satisfies: $p(\alpha_0|K, N, a, b) \propto p(K|\alpha_0, N)p(\alpha_0|a, b)$. We use a Metropolis-Hastings sampler to sample α_0. If the proposal distribution is symmetrical (e.g., a Gaussian distribution with mean α_0), then the accept ratio is:

$$A(\alpha_0, \alpha_0') = \frac{p(\alpha_0'|K, N, a, b)}{p(\alpha_0|K, N, a, b)} = \frac{p(K|\alpha_0', N)p(\alpha_0'|a, b)}{p(K|\alpha_0, N)p(\alpha_0|a, b)} \tag{4.4}$$

Examination of the likelihood function (4.3) reveals that its shape is similar to the Gamma distribution. For example, Figure 4.2 shows the normalized likelihood for α_0 in red, and a Gamma distribution fit to the same mean and variance in blue, for $N = 200$ and

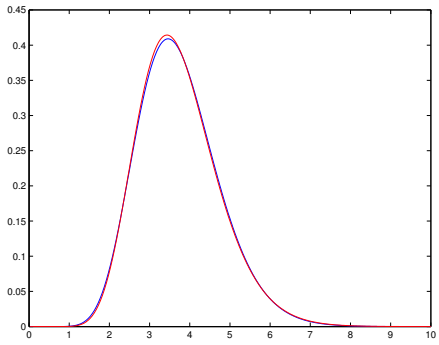


Figure 4.2: Gamma Distribution Fits Likelihood of α_0

$K = 15$. This similarity can be exploited to improve the mobility of the Metropolis-Hastings sampler. A Gamma proposal distribution can be chosen to approximate the posterior distribution of α_0 given K and N. Parameters $a(K, N)$ and $b(K, N)$ are estimated offline to match the mean and variance of the posterior distribution for different values of K and N. During sampling, proposal distribution parameters are selected to match the sample size N and the current number of consensus clusters K. The accept ratio for the Gamma proposal distribution is:

$$A(\alpha_0, \alpha_0') = \frac{p(K|\alpha_0', N)p(\alpha_0'|a, b)p(\alpha_0|a(K, N), b(K, N))}{p(K|\alpha_0, N)p(\alpha_0|a, b)p(\alpha_0'|a(K, N), b(K, N))} \tag{4.5}$$

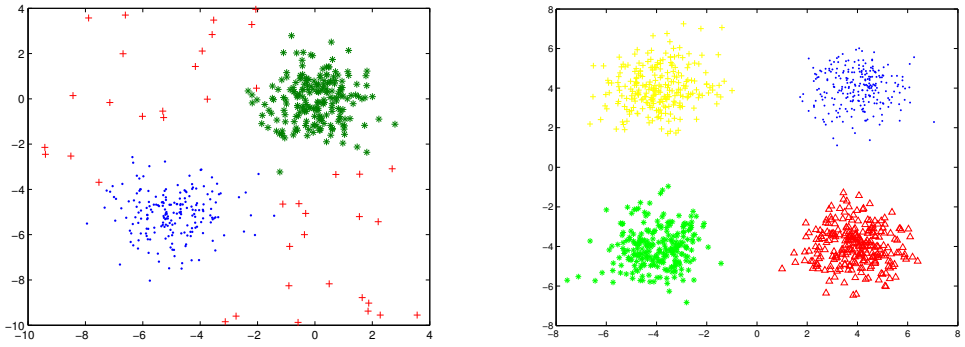
where $p(\cdot|a(K, N), b(K, N))$ is the Gamma distribution with its parameters $a(K, N)$ and $b(K, N)$ estimated by fitting the posterior distribution of α_0 given N and K.

4.3 Empirical Evaluation

We compared DPCE with two generative clustering ensemble models, Bayesian clustering ensembles (BCE) [84] and mixture model for clustering ensembles (MM) [77].

Datasets.

We evaluated DPCE on both synthetic and real datasets. First we generated two sets of synthetic data to test the robustness and accuracy of DPCE. The two synthetic datasets are plotted in Figure 4.3(a) and 4.3(b). The synthetic data shown in Figure 4.3(a) is consisted of two 2-dimensional Gaussian clusters (green stars and blue dots), each 75 points, with 50 uniform noise points (red pluses). The synthetic data shown in Figure 4.3(b) is consisted of four 2-dimensional Gaussian clusters without noise (yellow pluses, blue dots, green starts, and red triangles).



(a) Synthetic Dataset 1: Two Clusters with Outliers

(b) Synthetic Dataset 2: Four Clusters

Figure 4.3: Synthetic Datasets for DPCE

Then we used five datasets from the UCI Machine Learning Repository[2] to evaluate DPCE: *Glass*, *Ecoli*, *ImageSegmentation*, *ISOLET*, and *LetterRecognition*. *Glass* contains glass instances described by their chemical components. *Ecoli* contains data on E. Coli bacteria. *ImageSegmentation* contains data from images that were hand-segmented classifying each pixel. *ISOLET* contains data representing spoken letters of the alphabet. *LetterRecognition* contains character images corresponding to the capital letters in the English alphabet. We held out 1/4 of the data to evaluate the predictive performance of MM, BCE and DPCE.

[2]http://archive.ics.uci.edu/ml/

4.3.1 Methodology

For the synthetic datasets in Figure 4.3(a) and Figure 4.3(b), we used k-means to generate base clusterings. We varied the number of base clusterings and the number of clusters in each base clustering to test the robustness of DPCE.

To generate base clusterings on real data for each ensemble method, we ran Dirichlet Process Clustering (DPC) [54] 5 times with different random initializations to generate 5 base clusterings. We compared DPCE with DPC, MM, and BCE on real datasets. Also, we repeat each ensemble method 5 times with different base clustering results. For the parametric clustering ensembles methods, MM and BCE, we set the number of output clusters equal to the actual number of classes, according to the ground truth.

As for generative models, we used perplexity to compare them. The perplexity of the observed data $\vec{X}$ is defined as [9]:

$$perp(\vec{X}) = \exp\left(-\frac{\log p(\vec{X})}{N}\right) \tag{4.6}$$

where N is the number of data points in $\vec{X}$. Clearly, the perplexity monotonically decreases with the log-likelihood. Thus, a lower perplexity value on the training data means that the model fits the data better, and a lower value on the test data means that the model can better explain unseen data. For the five real datasets, we report perplexity on both training and test sets.

4.3.2 Results

First, we tested the robustness of DPCE on the two synthetic datasets. The first synthetic dataset shown in Figure 4.3(a) has two clusters with some noise. We fed five base clusterings with 4, 8, 16, 32, and 32 clusters to DPCE, that the five base clusterings all overestimate the true number of clusters. DPCE can always find 2 consensus clusters. Therefore the five base clusterings must contain coherent information about the cluster structure of the data,

Table 4.2: Perplexity Results on Training data for Real Datasets

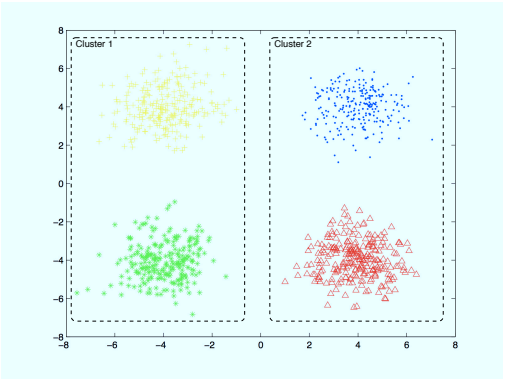
	Glass	Ecoli	ImageSegmentation	ISOLET	LetterRecognition
DPC	1.443 (0.22)	1.478 (0.41)	1.733 (0.042)	1.996 (0.46)	2.562 (0.051)
MM	1.323 (0.024)	1.465 (0.042)	1.704 (0.045)	1.986 (0.047)	2.536 (0.051)
BCE	1.093 (0.022)	1.320 (0.0.40)	1.545 (0.043)	1.874 (0.048)	2.248 (0.052)
DPCE	0.972 (0.023)	1.214 (0.0.43)	1.334 (0.044)	1.762 (0.047)	2.136 (0.051)

and DPCE can find that information. The second synthetic dataset shown in Figure 4.3(b) has four clusters, indicated by yellows pluses, green stars, blue dots and red triangles. We varied the number of base clusterings from 2, 4, 8, 16, to 32, with all base clusterings only have two clusters, which underestimate the true number of clusters. There are two types of base clusterings for the second synthetic dataset: the type 1, shown in Figure 4.4(a), groups yellow pluses and green stars together while blue dots and red triangles together, and the type 2, shown in Figure 4.4(b), groups yellow pluses and blue dots together while green stars and red triangles together. Further, no matter how many base clusterings used for the second synthetic dataset, we made one half of the base clusterings type 1, the other half type 2. DPCE can always find 4 consensus clusters from all base clusterings, shown in Figure 4.4(c). This demonstrates that DPCE is more robust and accurate than each individual base clustering, and more important, the number of consensus clusters is not bounded by the maximum and minimum number of clusters in base clusterings.

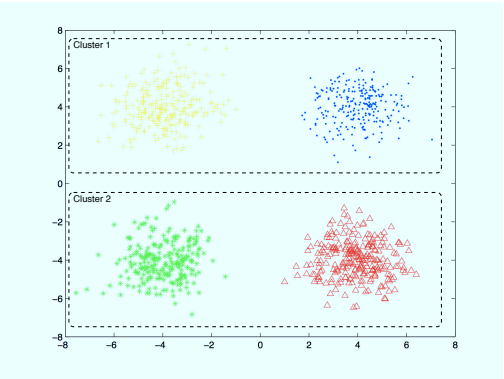
Table 4.2 compares DPC, MM, BCE and DPCE in terms of the perplexity on the synthetic datasets. It's clear that DPCE fits the data better than BCE, MM and DPC. BCE is better than MM. Both BCE and MM are better than DPC, but they are parametric models, and the number of consensus clusters is set according to the ground-truth. In contrast, DPCE can automatically find the number of clusters that fits the data best.

Tables 4.3 compare DPC, MM, BCE and DPCE in terms of the perplexity on test data of the real datasets. DPCE fits the test data best, and outperforms BCE, MM and DPC. BCE is better than MM, and BCE and MM are better than DPC.

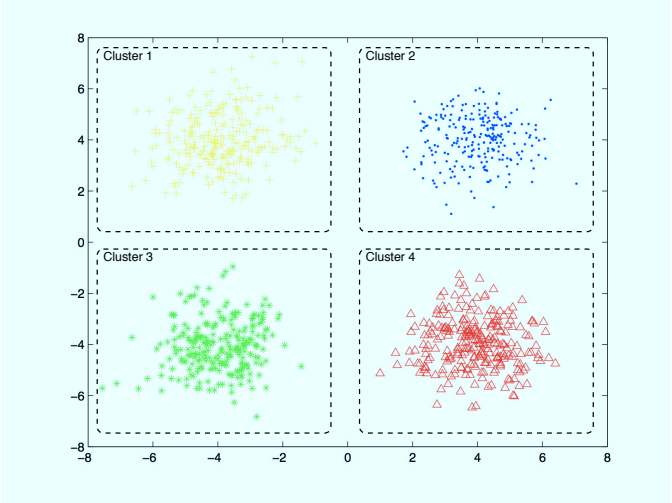
Figure 4.5 plots the log-likelihoods on the LetterRecognition dataset for 5 DPC runs and one DPCE run initialized with iteration 1000 of the 5 DPC runs. We continued the DPC runs for another 1000 iterations to compare with DPCE. All chains of DPCC and MPCC



(a) Base Clustering Type 1



(b) Base Clustering Type2



(c) Consensus Clustering

Figure 4.4: DPCE Results on Synthetic Dataset 2

appear to have reached different local optima, since the Potential Scale Reduction Factor MCMC diagnostic [19] for the 5 DPC log-likelihood values plotted in Figure 4.5 is 1.4908, which is indicative of non-convergence. The local optimum for DPCE has higher likelihood than all five DPC local optima.

Further, when MH sampling α_0, we used a Gamma proposal, which is evaluated by approximating the posterior distribution of α_0. To demonstrate the effectiveness of the Gamma proposal, we compared it with a Gaussian proposal, which proposes new α_0 distributed according to a Gaussian distribution centered at old α_0 with variance 1. We then

Table 4.3: Perplexity Results on Test Data for Real Datasets

	Glass	Ecoli	ImageSegmentation	ISOLET	LetterRecognition
DPC	1.215 (0.035)	1.994 (0.052)	2.501 (0.057)	2.620 (0.057)	3.774 (0.067)
MM	1.202 (0.036)	1.977 (0.051)	2.449 (0.056)	2.572 (0.058)	3.724 (0.069)
BCE	1.167 (0.035)	1.759 (0.049)	2.137 (0.059)	2.315 (0.055)	3.303 (0.068)
DPCE	1.025 (0.037)	1.524 (0.050)	1.933 (0.057)	2.014 (0.056)	2.956 (0.066)

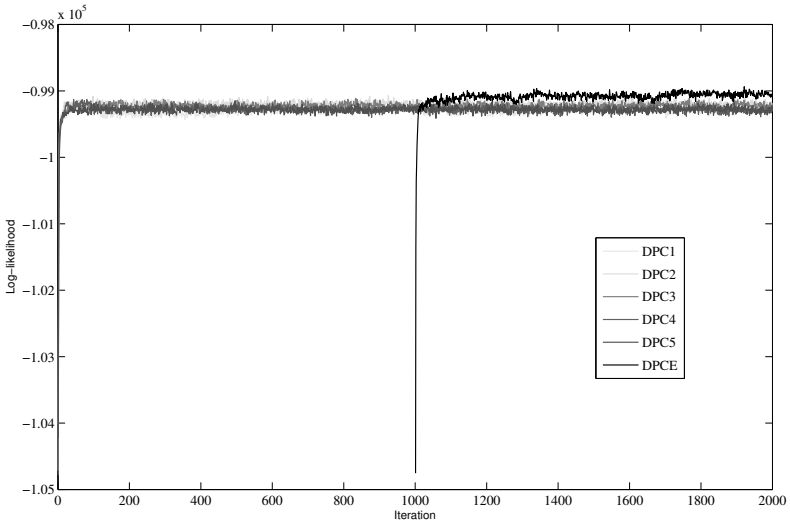


Figure 4.5: DPC and DPCE Likelihood Comparison

plot the correlation between samples proposed by Gamma proposal and Gaussian proposal respectively. Because Gaussian proposal can only propose local moves, the correlation between samples are very high; while Gamma proposal can propose non-local moves, the correlation between samples are relatively low. Figure 4.6 depicts the comparison of using Gamma proposal and Gaussian proposal when MH sampling α_0 on the synthetic dataset. Note we drop samples of the first half burn-in period, only plot the correlation between the second half samples.

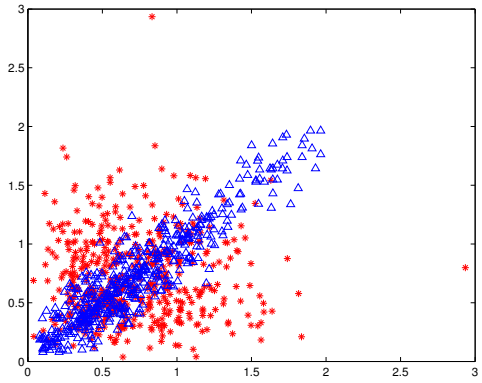


Figure 4.6: Correlation between samples of α_0 using Gaussian Proposal and Gamma proposal. Blue triangles are samples from Gaussian proposal, and red stars are from Gamma proposal.

Chapter 5: Nonparametric Bayesian Co-clustering Ensembles

5.1 Introduction

A direct extension to DPCE is to apply ensembles to *co-clustering*, the problem of simultaneously clustering the rows and columns of a data matrix into row- and column-clusters to achieve homogeneity in the blocks in the induced partition of the data matrix. Our first approach to co-clustering ensembles extends DPCE to a nonparametric model for co-clustering ensembles. While nonparametric Bayesian methods have previously been used in co-clustering [49] to allow the number of row clusters and column clusters to be random and inferred from the data, our work makes use of nonparametric Bayesian ideas to model co-clustering *ensembles*. In particular, I develop a model-based approach to ensembles that explicitly models the way in which multiple co-clusterings differ from each other and from a consensus co-clustering.

One way in which multiple co-clusterings can arise is via different local optima of a single base co-clustering method. Rather than selecting one of these optima, our approach explicitly recognizes the possibility that these local optima may contribute distinct, complementary perspectives on the co-clustering problem, in which case all optima should contribute to the formation of a consensus co-clustering. It is worth noting that this issue arises in many problems in which there is combinatorial structure, and our model-based approach to ensembles may have applications beyond co-clustering.

Most co-clustering algorithms [12, 69, 70, 85] assume that row- and column-clusters are variation independent; i.e., individual co-clusters are obtained as the product of row- and column-clusters. This partitions the data matrix into a regular grid. This assumption of variation independence is inappropriate in situations exhibiting context-specific independence. For example, one cannot represent the situation in which, for some rows, a given set a of

columns is partitioned into several clusters, whereas for other rows, the columns form a single undifferentiated cluster. Recent work has explored a nonparametric Bayesian prior known as the *Mondrian Process* that relaxes this assumption [65]. A sample drawn from a two-dimensional Mondrian process is a random partition over a matrix that is not constrained to be a regular grid. Our second approach to co-clustering ensembles is based on Mondrian processes.

Specifically I develop (1) a Dirichlet process-based co-clustering ensemble model (DPCCE), which assumes independent Dirichlet process mixture priors for rows and columns; and (2) a Mondrian process-based co-clustering ensemble model (MPCCE) that places a Mondrian process prior over the matrix partitions. For both the DPCCE and the MPCCE, the number of blocks is not fixed *a priori*, but is open-ended and inferred from the data.

5.2 Dirichlet Process-based Co-clustering Ensembles

Following general practice in the clustering ensemble literature, [71], the DPCCE model does not specify a probabilistic model for the original $R \times C$ data matrix $\vec{X}$, but rather models the output of M base co-clusterings $\langle \varphi_m | m \in \{1, 2, \cdots, M\} \rangle$. The base co-cluster φ_m partitions the rows and columns of the data matrix into I_m row clusters and J_m column clusters. We assume that rows and columns are clustered independently by the base clusterings, resulting in a grid-style partition. That is, all entries in a given row (column) are assigned to the same row (column) cluster. The base co-clusterings are organized into a $R \times C \times M$ array $\vec{Y}$, where the entries $y_{rcm} = \langle y^R_{rm}, y^C_{cm} \rangle$ denote the row- and column-cluster ID's assigned by φ_m. The indices y^R_{rm} and y^C_{cm} range from 1 to I_m and J_m, respectively.

5.2.1 DPCCE Generative Model

According to the DPCCE model, the observations $\vec{Y}$ are generated from independent row and column Dirichlet process mixture models with pseudo-counts α^R and α^C, and row and column base measures G^R_m and G^C_m, respectively. Figure 5.1 depicts the DPCCE model.

A stick-breaking process is used to generate the row and column Dirichlet processes. The mixing proportions $\vec{\pi}^R$ and $\vec{\pi}^C$ are generated as in Eq (2.5), and the consensus cluster indicator variables z_r^R and z_c^C are drawn according to these mixing proportions. The unique row and column parameters $\vec{\theta}_{lm}^{*R}$ and $\vec{\theta}_{km}^{*C}$ for each consensus row-cluster l and column-cluster k are generated as independent draws from symmetric T-dimensional Dirichlet distributions G_m^R and G_m^C with pseudo-counts β_m^R and β_m^C, respectively. We assume $I_m, J_m \leq T$; as T grows without bound with fixed total pseudo-count, G_m^R and G_m^C become Dirichlet process distributions. The row-cluster ID's y_{rm}^R are independent draws from a T-dimensional discrete distribution with parameter $\vec{\theta}_{lm}^{*R}$, where $l = z_r^R$ is the row-cluster indicator for row r. Similarly, the column-cluster ID's y_{cm}^C are independent draws from a T-dimensional discrete distribution with parameter $\vec{\theta}_{km}^{*C}$, where $k = z_c^C$ is the column-cluster indicator for row r.

Formally, the generative process for DPCCE is:

- Draw $v_l^R \sim \text{Beta}(1, \alpha^R)$, for $l = 1, 2, \cdots, \infty$
- Set mixture weights for consensus row-clusters $\pi_l^R = v_l^R \prod_{t=1}^{l-1}(1 - v_t^R)$, for $l = 1, 2, \cdots, \infty$
- Draw $v_k^C \sim \text{Beta}(1, \alpha^C)$, for $k = 1, 2, \cdots, \infty$
- Set mixture weights for consensus column-clusters $\pi_k^C = v_k^C \prod_{t=1}^{k-1}(1 - v_t^C)$, for $k = 1, 2, \cdots, \infty$
- Draw parameters for consensus row-clusters $\vec{\theta}_l^{*R} \sim \text{Dir}(\beta^R)$, for $l = 1, 2, \cdots, \infty$
- Draw parameters for consensus column-clusters $\vec{\theta}_k^{*C} \sim \text{Dir}(\beta^C)$, for $k = 1, 2, \cdots, \infty$
- For each row r:
  - Draw consensus row-cluster $z_r^R \sim \text{Discrete}(\vec{\pi}^R)$
  - For each base co-clustering φ_m:
    - Generate $y_{rm}^R \sim \text{Discrete}(\vec{\theta}_{lm}^{*R})$, where $l = z_r^R$

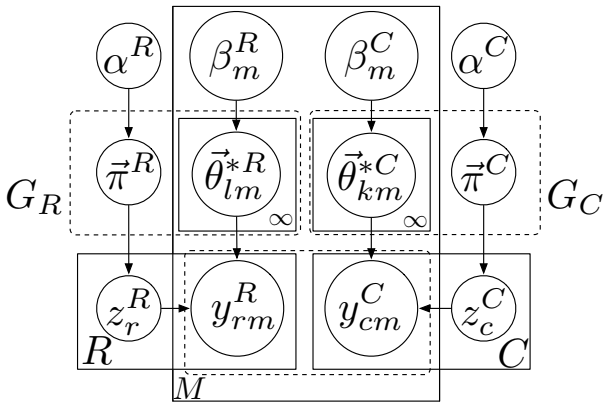


Figure 5.1: Dirichlet Process-based Co-clustering Ensemble Model

- For each column c:
  - Draw consensus column-cluster $z_c^C \sim \text{Discrete}(\vec{\pi}^C)$
  - For each base co-clustering φ_m:
    - Generate $y_{cm}^C \sim \text{Discrete}(\vec{\theta}^{*C}_{km})$, where $k = z_c^C$

5.2.2 Inference

We use the collapsed Gibbs sampling method discussed in Sec. 2.3.1 for DPCCE inference. As all model parameters are marginalized out, we sample only z_r^R and z_c^C. We assume infinite T, so that G_m^R and G_m^C become Dirichlet process distributions.

The conditional distribution for sampling z_r^R given $\vec{Y}$ and all other indicator variables $\vec{z}^{R\neg r}$ is:

$$p(z_r^R = l|\vec{Y}, \vec{z}^{R\neg r}, \gamma^R) \propto \tag{5.1}$$

$$\frac{\mathcal{N}_l^{R\neg r}}{R - 1 + \alpha^R} \prod_{m=1}^{M} \mathcal{N}_{y_{rm}^R}^{R\neg r}$$

when the cluster index l appears among the indices in $\vec{z}^{R\neg r}$, and

$$p(z_r^R = l|\vec{Y}, \vec{z}^{R\neg r}, \gamma^R) \propto \frac{\alpha^R}{R-1+\alpha^R} \prod_{m=1}^{M} \mathcal{N}_{y_{rm}^R}^{R^{\neg r}} \tag{5.2}$$

when the cluster index l does not appear among the indices in $\vec{z}^{R\neg r}$. Here, $\mathcal{N}_l^{R^{\neg r}}$ is the number of rows assigned to the l^{th} consensus row-cluster excluding the r^{th} row, and $\mathcal{N}_{y_{rm}^R}^{R^{\neg r}}$ is the number rows assigned to the same row-cluster as the r^{th} row by φ_m excluding the r^{th} row.

Similarly, the conditional distribution for sampling z_c^C given $\vec{Y}$ and all other indicator variables $\vec{z}^{C\neg c}$ is:

$$p(z_c^C = k|\vec{Y}, \vec{z}^{C\neg c}, \gamma^C) \propto \frac{\mathcal{N}_k^{C^{\neg c}}}{C-1+\alpha^C} \prod_{m=1}^{M} \mathcal{N}_{y_{cm}^C}^{C^{\neg c}} \tag{5.3}$$

when the cluster index k appears among the indices in $\vec{z}^{C\neg c}$, and

$$p(z_c^C = k|\vec{Y}, \vec{z}^{C\neg c}, \gamma^C) \propto \frac{\alpha^C}{C-1+\alpha^C} \prod_{m=1}^{M} \mathcal{N}_{y_{cm}^C}^{C^{\neg c}} \tag{5.4}$$

when the cluster index k does not appear among the indices in $\vec{z}^{C\neg c}$. Here, $\mathcal{N}_k^{C^{\neg c}}$ is the number of columns assigned to the k^{th} consensus column-cluster excluding the c^{th} column, and $\mathcal{N}_{y_{cm}^C}^{C^{\neg c}}$ is the number columns assigned to the same column-cluster as the c^{th} column by φ_m excluding the c^{th} column.

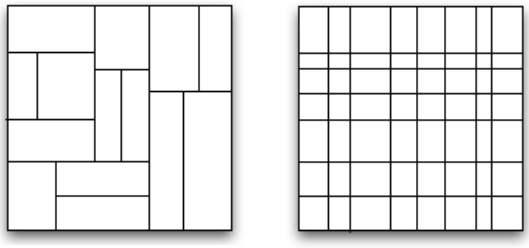


Figure 5.2: Unpermuted Synthetic Data Matrix Sampled from Mondrian Process (left) and Corresponding Grid (right)

Table 5.1 summarizes notation used throughout the paper.

5.3 Mondrian Process-based Co-clustering Ensembles

The Mondrian Process-based Co-clustering Ensemble (MPCCE) model generalizes the grid-style partitions of the DPCCE to allow different resolutions in different parts of the data matrix. The non-regular partitions generated by the MP provide increased flexibility and parsimony.

A sample drawn from a two-dimensional Mondrian Process partitions a rectangle using axis-aligned cuts, as illustrated in Figure 5.2 (left). If we overlay this partition on a data matrix, we can identify each block with a co-cluster consisting of entries falling inside the block. The model replaces the independent row clusters and column clusters of the DPCCE model with a set of co-clusters. It is more natural to deal with these co-clusters directly, rather than with row- and column-clusters separately. To achieve the same level of resolution with a grid-style partition would require a much less parsimonious model, as shown in Figure 5.2 (right).

5.3.1 MPCCE Generative Model

The MPCCE generative process, depicted in Figure 5.3, puts a two-dimensional MP prior on partitions of the data matrix. Following [65], we treat a MP prior as generating a partition $\mathcal{M}$ over the unit square $[0, 1] \times [0, 1]$. Rows and columns of the data matrix are mapped to

Table 5.1: Notation Description for DPCCE and MPCCE

Symbols	Description
R	number of rows in the data matrix $\vec{X}$
C	number of columns in the data matrix $\vec{X}$
M	number of base co-clusterings
φ_m	the m^{th} base co-clustering
	Notation for DPCCE
I_m	number of row-clusters in φ_m
J_m	number of column-clusters in φ_m
y^R_{rm}	the row-cluster assigned to the r^{th} row by φ_m, $y^R_{rm} \in \{1, \cdots, I_m\}$
y^C_{cm}	the column-cluster assigned to the c^{th} column by φ_m, $y^C_{cm} \in \{1, \cdots, J_m\}$
$\vec{Y}$	defined as $\langle y_{rcm} \vert r \in \{1, \cdots, R\}, c \in \{1, \cdots, C\}, m \in \{1, \cdots, M\}\rangle$
$\vec{\theta}^{R*}_{lm}$	the discrete distribution of observing the row-clusters of φ_m in the l^{th} consensus row-cluster
$\vec{\theta}^{C*}_{km}$	the discrete distribution of observing the column-clusters of φ_m in the k^{th} consensus column-cluster
$\vec{\theta}^{C*}_{l}$	defined as $\langle \theta^{R*}_{lm} \vert m \in \{1, 2, \cdots, M\}\rangle$
$\vec{\theta}^{C*}_{k}$	defined as $\langle \theta^{C*}_{km} \vert m \in \{1, 2, \cdots, M\}\rangle$
$\mathcal{N}^R_{i_m}$	the number of rows assigned to the ${i_m}^{th}$ row-cluster by φ_m
$\mathcal{N}^C_{j_m}$	the number of columns assigned to the ${j_m}^{th}$ column-cluster by φ_m
$\mathcal{N}^R_{l}$	the number of rows assigned to the l^{th} consensus row-cluster
$\mathcal{N}^C_{k}$	the number of columns assigned to the k^{th} consensus column-cluster
$\mathcal{N}^{R^{\neg r}}_{l}$	the number of rows assigned to the l^{th} consensus row-cluster excluding the r^{th} row
$\mathcal{N}^{C^{\neg c}}_{k}$	the number of columns assigned to the k^{th} consensus column-cluster excluding the c^{th} column
$\mathcal{N}^{R^{\neg r}}_{y^R_{r\cdot m}}$	the number rows assigned to the same row-cluster as the r^{th} row by φ_m excluding the r^{th} row
$\mathcal{N}^{C^{\neg c}}_{y^C_{\cdot cm}}$	the number of columns assigned to the same column-cluster of the c^{th} column by φ_m, excluding the c^{th} column
	Notation for MPCCE
J_m	number of co-clusters in φ_m
$\mathcal{M}$	a Mondrian sample, which is a Mondrian style partition over the unit square, and assume there are K blocks in $\mathcal{M}$
y_{rcm}	the co-cluster identity assigned to the entry (r, c) by the m^{th} base clustering φ_m, $y_{rcm} \in \{1, \cdots, K\}$
$\vec{Y}$	defined as $\langle y_{rcm} \vert r \in \{1, \cdots, R\}, c \in \{1, \cdots, C\}, m \in \{1, \cdots, M\}\rangle$
θ_{mkj_m}	the probability of assigning an entry in the k^{th} block of $\mathcal{M}$ by φ_m to its ${j_m}^{th}$ co-cluster
$\vec{\theta}_{mk}$	defined as $\langle \theta_{mkj_m} \vert j_m \in \{1, 2, \cdots, J_m\}\rangle$, which is drawn from a J_m-dimensional symmetric Dirichlet distribution with hyperparameter β_m
χ^R_h	the position of the h^{th} horizontal cut of the total L_R horizontal cuts in $\mathcal{M}$
χ^C_g	the position of the g^{th} vertical cut of the total L_C vertical cuts in $\mathcal{M}$
$\mathcal{N}_k$	the number of entries in the k^{th} block of $\mathcal{M}$
$\mathcal{N}^{y_{\cdot\cdot m}=j_m}_k$	the number of entries in both the k^{th} block of $\mathcal{M}$ and the ${j_m}^{th}$ co-cluster of φ_m
$\mathcal{N}^{\neg r}_k$	the number of entries in the k^{th} block of $\mathcal{M}$, excluding the entries in the r^{th} row
$\mathcal{N}^{\neg c}_k$	the number of entries in the k^{th} block of $\mathcal{M}$, excluding the entries in the c^{th} column
$\mathcal{N}^{\neg r}_{k,y_{\cdot\cdot m}=j_m}$	the number of entries in both the k^{th} block of $\mathcal{M}$ and the ${j_m}^{th}$ co-cluster of φ_m, excluding the entries in the r^{th} row
$\mathcal{N}^{\neg c}_{k,y_{\cdot\cdot m}=j_m}$	the number of entries in both the k^{th} block of $\mathcal{M}$ and the ${j_m}^{th}$ co-cluster of φ_m, excluding the entries in the c^{th} column

vertical and horizontal coordinates of the unit square through latent variables ξ_r and η_c. The latent variables $\vec{\xi} = \langle \xi_r | r \in \{1, \cdots, R\} \rangle$ and $\vec{\eta} = \langle \eta_c | c \in \{1, \cdots, C\} \rangle$ act like permutations of the rows and columns of the data matrix. The partition $\mathcal{M}$ and the latent variables $\vec{\xi}$ and $\vec{\eta}$ determine a partition over the original data matrix.

As with DPCCE and standard practice in the clustering ensemble literature and model the variables y_{rcm} that denote the co-cluster ID assigned to the entry in row r and column c by the m^{th} base clustering φ_m. The co-cluster ID y_{rcm} ranges from 1 to J_m, the number of co-clusters output by φ_m. We assume that y_{rcm} is sampled from a discrete distribution with parameter $\vec{\theta}_{mk}$, namely $p(y_{rcm} = j_m) = \theta_{mkj_m}$, where k is the block of $\mathcal{M}$ corresponding to row r and column c, and the parameter $\vec{\theta}_{mk}$ is sampled from a symmetric J_m-dimensional Dirichlet distribution.

Formally, the generative process for the base clusterings $\vec{Y}$ proceeds as follows:

- Draw a partition $\mathcal{M} \sim MP(\lambda, [0,1], [0,1])$; let K be the number of blocks in $\mathcal{M}$
- Draw block parameters $\vec{\theta}_{mk} \sim \text{Dir}(\beta_m)$, for $m = 1, 2, \cdots, M$ and $k = 1, 2, \cdots, K$
- Draw latent row coordinates $\xi_r \sim \text{Uniform}[0,1]$, for $r = 1, 2, \cdots, R$
- Draw latent column coordinates $\eta_c \sim \text{Uniform}[0,1]$, for $c = 1, 2, \cdots, C$
- For each row r and column c:
  - Let k be the block (co-cluster) of $\mathcal{M}$ to which (ξ_r, η_c) belongs
  - For each base clustering φ_m, draw $y_{rcm} \sim \text{Discrete}(\vec{\theta}_{mk})$

5.3.2 Inference

We perform Markov Chain Monte Carlo (MCMC) simulation on the posterior distribution over $\mathcal{M}$, $\vec{\xi}$, $\vec{\eta}$, and $\vec{\theta}$. The joint distribution of observed base co-clustering results $\vec{Y}$, hidden

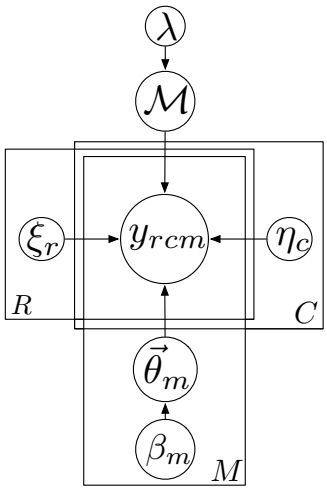


Figure 5.3: Mondrian Process-based Co-clustering Ensemble Model

variable $\mathcal{M}$, $\vec{\xi}$ and $\vec{\eta}$, and model parameters $\vec{\theta}$ is:

$$p(\vec{Y}, \mathcal{M}, \vec{\xi}, \vec{\eta}, \vec{\theta}|\beta, \lambda) = p(\mathcal{M}|\lambda) \left(\prod_{r=1}^{R} p(\xi_r) \right) \left(\prod_{c=1}^{C} p(\eta_c) \right) \tag{5.5}$$

$$\left(\prod_{k=1}^{K} \prod_{m=1}^{M} p(\vec{\theta}_{mk}|\beta) \right) \left(\prod_{r=1}^{R} \prod_{c=1}^{C} \prod_{m=1}^{M} p(y_{rcm}|\vec{\theta}, \mathcal{M}, \xi_r, \eta_c) \right).$$

We can integrate out the model parameter $\vec{\theta}$ because of conjugacy:

$$p(\vec{Y}, \mathcal{M}, \vec{\xi}, \vec{\eta}|\beta, \lambda) = p(\mathcal{M}|\lambda) \left(\prod_{r=1}^{R} p(\xi_r) \right) \left(\prod_{c=1}^{C} p(\eta_c) \right) \tag{5.6}$$

$$\left(\prod_{k=1}^{K} \prod_{m=1}^{M} \frac{\Gamma(J_m \beta_m)}{\Gamma(J_m \beta_m + \mathcal{N}_k)} \prod_{j_m=1}^{J_m} \frac{\Gamma(\beta_m + \mathcal{N}_k^{y_{\cdot\cdot m}=j_m})}{\Gamma(\beta_m)} \right),$$

where $\mathcal{N}_k$ denotes the number of entries in the k^{th} block of $\mathcal{M}$, and $\mathcal{N}_k^{y_{\cdot\cdot m}=j_m}$ denotes the number of entries in both the k^{th} block of $\mathcal{M}$ and the ${j_m}^{th}$ co-cluster of φ_m.

We perform Gibbs sampling on the row and column coordinates $\vec{\xi}$ and $\vec{\eta}$. Since ξ_r and

η_c have uniform prior distributions, their posterior distributions are piece-wise constant [65]. Define $\vec{\chi}^R = \langle \chi_h^R | h \in \{0, \cdots, L_R, L_R+1\} \rangle$, where $\chi_0^R = 0$, $\chi_h^R < \chi_{h+1}^R$, $\chi_{L_R+1}^R = 1$. The value χ_h^R is the position of the h^{th} horizontal cut of the total L_R horizontal cuts in $\mathcal{M}$. The conditional probability that ξ_r falls in the interval (χ_h^R, χ_{h+1}^R) is:

$$p(\chi_h^R < \xi_r < \chi_{h+1}^R | \vec{X}, \mathcal{M}, \vec{\xi}^{\neg r}, \vec{\eta}, \beta, \lambda) \propto \tag{5.7}$$
$$(\chi_{h+1}^R - \chi_h^R) \Big(\prod_{k=1}^{K} \prod_{m=1}^{M} \frac{\Gamma(J_m \beta_m)}{\Gamma(J_m \beta_m + \mathcal{N}_k^{\neg r})} \prod_{j_m=1}^{J_m} \frac{\Gamma(\beta_m + \mathcal{N}_{k,y_{\cdot\cdot m}=j_m}^{\neg r})}{\Gamma(\beta_m)} \Big).$$

Similarly, let $\vec{\chi}^C = \langle \chi_g^C | g \in \{0, \cdots, L_C, L_C+1\} \rangle$, where $\chi_0^C = 0$, $\chi_g^C < \chi_{g+1}^C$, $\chi_{L_C+1}^C = 1$. The value χ_g^C is the position of the g^{th} vertical cut of the total L_C vertical cuts in $\mathcal{M}$. The conditional probability that η_c falls in the interval (χ_g^C, χ_{g+1}^C) is:

$$p(\chi_g^C < \eta_c < \chi_{g+1}^C | \vec{X}, \mathcal{M}, \vec{\xi}, \vec{\eta}^{\neg c}, \beta, \lambda) \propto \tag{5.8}$$
$$(\chi_{g+1}^C - \chi_g^C) \Big(\prod_{k=1}^{K} \prod_{m=1}^{M} \frac{\Gamma(J_m \beta_m)}{\Gamma(J_m \beta_m + \mathcal{N}_k^{\neg c})} \prod_{j_m=1}^{J_m} \frac{\Gamma(\beta_m + \mathcal{N}_{k,y_{\cdot\cdot m}=j_m}^{\neg c})}{\Gamma(\beta_m)} \Big).$$

In these equations, the superscripts $\neg r$ and $\neg c$ mean that the r^{th} row and c^{th} column are excluded in the respective counts. Accordingly, we have:

$$\theta_{mkj_m} \propto \beta_m + \mathcal{N}_k^{y_{\cdot\cdot m}=j_m}. \tag{5.9}$$

Reversible jump MCMC (RJMCMC) [22] is used to sample from the posterior distribution $p(\mathcal{M} | \vec{Y}, \vec{\xi}, \vec{\eta}, \beta, \lambda)$. A state $\mathcal{M}$ consists of a tree of blocks and a vector $\vec{\zeta}$ of parameters. The parameters consist of a cost E_k and a location χ_k of the cut to each non-leaf block of the tree. The location χ_k ranges between zero and τ_k, where τ_k is half the length of the block perimeter. If χ_k is less than the width of the block, a vertical cut is made at position χ_k

along the width; otherwise, a horizontal cut is made along the height of the block at position equal to χ_k minus the block width.

Each MCMC proposal either removes a pair of sibling leaf blocks or adds a cut to a leaf block. When a leaf block k is split into child blocks k' and k'', the parameter $\vec{\zeta}$ is extended to $\langle \vec{\zeta}, E_k, \chi_k \rangle$. When a split is removed, the associated cost E_k and location χ_k are removed from $\langle \vec{\zeta}, E_k, \chi_k \rangle$ to obtain $\vec{\zeta}$. RJMCMC maintains reversibility of moves by adding auxiliary parameters so that moves occur between spaces of equal dimensions. When proposing to add a cut, we augment the current parameter $\vec{\zeta}_t$ and define a bijection between the augmented parameter $\langle \vec{\zeta}_t, u_1, u_2 \rangle$ and the proposed parameter $\vec{\zeta}_{t+1} = \langle \vec{\zeta}_t, E_k, \chi_k \rangle$:

$$g^{add}_{t \to t+1}(\langle \vec{\zeta}_t, u_1, u_2 \rangle) = \langle \vec{\zeta}_t, E_k, \chi_k \rangle. \tag{5.10}$$

Similarly, when proposing to remove a cut, we augment the proposed state $\vec{\zeta}_{t+1}$ and define a bijection between the current state $\vec{\zeta}_t$ and the augmented proposed state $\langle \vec{\zeta}_{t+1}, u_1, u_2 \rangle$:

$$g^{remove}_{t \to t+1}(\vec{\zeta}_t) = \tag{5.11}$$
$$g^{remove}_{t \to t+1}(\langle \vec{\zeta}_{t+1}, E_k, \chi_k \rangle) = \langle \vec{\zeta}_{t+1}, u_1, u_2 \rangle.$$

The proposal distribution $Q(\mathcal{M}_{t+1}; \mathcal{M}_t)$ chooses with equal probability whether to add or remove a cut, and uses a uniform discrete distribution to sample the block at which to add or remove the cut. When a cut at block k is being added, $Q(\mathcal{M}_{t+1}; \mathcal{M}_t)$ proposes a location χ_k from a uniform distribution and a cost E_k from an exponential distribution with parameter τ_k. When a cut at block k is being removed, $Q(\mathcal{M}_{t+1}; \mathcal{M}_t)$ sets the new parameter $\vec{\zeta}_{t+1}$ deterministically by removing the cost E_k and location χ_k from the current state $\vec{\zeta}_t$, and the auxiliary parameters are then sampled from a distribution $q(u_1, u_2)$. The parameter u_1 is sampled from the same exponential distribution used to sample the cost of a new cut at k, and the parameter u_2 is sampled from the same uniform distribution used

to sample the location of a new cut at k.

Following [22], the proposal to remove a cut is accepted if α drawn from $Uniform(0,1)$ satisfies:

$$\alpha < \min\Bigg\{1, \frac{p(\mathcal{M}_{t+1}|\vec{Y},\vec{\xi},\vec{\eta},\beta,\lambda)}{p(\mathcal{M}_t|\vec{Y},\vec{\xi},\vec{\eta},\beta,\lambda)} \frac{Q(\mathcal{M}_t;\mathcal{M}_{t+1})}{Q(\mathcal{M}_{t+1};\mathcal{M}_t)q(u_1,u_2)} \left|\frac{\partial\langle\vec{\zeta}_{t+1},u_1,u_2\rangle}{\partial\vec{\zeta}_t}\right|\Bigg\}, \tag{5.12}$$

where $\left|\frac{\partial\langle\vec{\zeta}_{t+1},u_1,u_2\rangle}{\partial\vec{\zeta}_t}\right|$ is the Jacobian of $g^{remove}_{t\to t+1}(\vec{\zeta}_t)$. The acceptance probability for adding a cut is obtained in a similar manner. See [22] for details on RJMCMC.

To calculate the acceptance ratio in Equation (5.12), we need to calculate two ratios $\frac{Q(\mathcal{M}_t;\mathcal{M}_{t+1})}{Q(\mathcal{M}_{t+1};\mathcal{M}_t)q(\vec{U}_{t+1})}$ and $\frac{p(\mathcal{M}_{t+1}|\vec{Y},\vec{\xi},\vec{\eta},\beta,\lambda)}{p(\mathcal{M}_t|\vec{Y},\vec{\xi},\vec{\eta},\beta,\lambda)}$. The first of these involves only the proposal distributions, and is straightforward to calculate. The second of these, the ratio of posterior probabilities of $\mathcal{M}_{t+1}$ and $\mathcal{M}_t$, is equal to the prior odds ratio times the likelihood ratio:

$$\frac{p(\mathcal{M}_{t+1}|\vec{Y},\vec{\xi},\vec{\eta},\beta,\lambda)}{p(\mathcal{M}_t|\vec{Y},\vec{\xi},\vec{\eta},\beta,\lambda)} = \frac{p(\mathcal{M}_{t+1}|\lambda)}{p(\mathcal{M}_t|\lambda)} \frac{\mathcal{L}(\mathcal{M}_{t+1})}{\mathcal{L}(\mathcal{M}_t)}, \tag{5.13}$$

where $\mathcal{L}(\mathcal{M}_{t+1})$ and $\mathcal{L}(\mathcal{M}_t)$ are the likelihood of $\mathcal{M}_{t+1}$ and $\mathcal{M}_t$, which are defined as:

$$\mathcal{L}(\mathcal{M}_{t+1}) = \tag{5.14}$$

$$\prod_{k_{t+1}=1}^{K_{t+1}} \prod_{m=1}^{M} \frac{\Gamma(J_m\beta_m)}{\Gamma(J_m\beta_m + \mathcal{N}_{k_{t+1}})} \prod_{j_m=1}^{J_m} \frac{\Gamma(\beta_m + \mathcal{N}_{k_{t+1}}^{y_{\cdot\cdot m}=j_m})}{\Gamma(\beta_m)},$$

$$\mathcal{L}(\mathcal{M}_t) = \tag{5.15}$$

$$\prod_{k_t=1}^{K_t} \prod_{m=1}^{M} \frac{\Gamma(J_m\beta_m)}{\Gamma(J_m\beta_m + \mathcal{N}_{k_t})} \prod_{j_m=1}^{J_m} \frac{\Gamma(\beta_m + \mathcal{N}_{k_t}^{x_{\cdot\cdot m}=j_m})}{\Gamma(\beta_m)}.$$

For a proposal to remove a cut of block k into blocks k' and k'', the prior odds ratio is given by:

$$\frac{p(\mathcal{M}_{t+1}|\lambda)}{p(\mathcal{M}_t|\lambda)} = \frac{\omega_k}{p(\chi_k)p(E_k)\omega_{k'}\omega_{k''}}, \tag{5.16}$$

where ω_k is the probability that sampling terminates with no cut at block k; this happens when the cost E_k exceeds the budget λ_k. The cut cost E_k is generated from an exponential distribution with parameter τ_k. Thus, the probability of terminating with no split at block k is given by:

$$\omega_k = \tag{5.17}$$

$$\int_{\lambda_k}^{+\infty} \tau_k \exp(-\tau_k e) de = \exp(-\tau_k \lambda_k).$$

Similarly, $\omega_{k'} = \exp(-\tau_{k'}\lambda_{k'})$ and $\omega_{k''} = \exp(-\tau_{k''}\lambda_{k''})$. Note that a block's budget is equal to its parent's budget minus the cost of cutting the parent. Thus, $\lambda'_k = \lambda''_k = \lambda_k - E_k$; and λ_k can be computed recursively from the budgets and cut costs of its ancestors.

A similar calculation gives the acceptance ratio for adding a random cut to $\mathcal{M}_t$ to generate $\mathcal{M}_{t+1}$. The inference algorithm for MPCCE is given in Algorithm 3.

Algorithm 3 Inference for MPCCE

Input λ, β and $\vec{Y}$; randomly initialize $\vec{\xi}$ and $\vec{\eta}$
$t \leftarrow 0$
$\mathcal{M}_0$ has no cut
budget$\leftarrow \lambda$
repeat
  $t \leftarrow t + 1$
  Propose $\mathcal{M}_{t+1}$ conditioned on $\mathcal{M}_t$ by either adding or removing a cut
  Accept or reject $\mathcal{M}_{t+1}$ according to Equation (5.12)
  if reject **then**
    $\mathcal{M}_{t+1} \leftarrow \mathcal{M}_t$
  else
    $\mathcal{M}_{t+1} \leftarrow \mathcal{M}_{t+1}$
  end if
  Gibbs sample $\vec{\xi}$ and $\vec{\eta}$ according to Equation (5.7) and (5.8)
until Stopping criteria met
Output the final $\mathcal{M}$, $\vec{\xi}$ and $\vec{\eta}$

5.4 Empirical Evaluation

We compared DPCCE and MPCCE with other generative co-clustering approaches: Latent Dirichlet Co-clustering (LDCC) [70, 85], Dirichlet Process-based Co-clustering (DPCC) [49], and Mondrian Process-based Co-clustering (MPCC) [65].

5.4.1 Data

We conducted evaluation of DPCCE and MPCCE on both synthetic and real data. Following [65], we synthetically generated non grid-style clusters by sampling from a Mondrian process on the unit square. We then generated 250 row and 250 column coordinates from a uniform distribution, and set the data value to the cluster ID for the block at those coordinates. Finally, we permuted the rows and columns randomly to form the final data matrix. We also used two real datasets for DPCCE and MPCCE: (a) MovieLens[1] is a movie recommendation dataset containing 100,000 ratings in a sparse data matrix for 1682 movies rated by 943 users. (b) Jester[2] is a joke rating dataset. The original dataset contains 4.1 million continuous ratings of 100 jokes from 73,421 users. Following [70], we chose 1000 users who rated almost

[1] http://www.grouplens.org/node/73
[2] http://goldberg.berkeley.edu/jester-data/

all jokes, discretized the ratings, and used this dense data matrix in our experiment. For both real datasets, we held out 25% of the data for testing.

5.4.2 Methodology

We compared DPCCE and MPCCE with other generative co-clustering approaches: Latent Dirichlet Co-clustering (LDCC) [70, 85], Dirichlet Process-based Co-clustering (DPCC) [49], and Mondrian Process-based Co-clustering (MPCC) [65]. LDCC requires specification of the numbers of row- and column-clusters. For the synthetic dataset, we varied the numbers of both row- and column-clusters from 5 to 10. For MovieLens, we set the number of user clusters to 20, the number of occupation categories, and the number of movie clusters to 19, the number of genres. For Jester, we used 5 joke clusters and 20 user clusters; this is the number of clusters given in the data description. The pseudo-counts of the DP priors for both rows and columns in DPCC and DPCCE are assumed a Gamma prior, as for DPCE. We ran DPCC and MPCC 5 times with different random initializations, to generate five base co-clustering results. We then ran DPCCE and MPCCE based on the DPCC and MPCC results, respectively. We repeated DPCCE and MPCCE 5 times, each time with five different base co-clusterings. For MPCCE and MPCC we set the budget $\lambda = 1$, and let μ_d be Lebesgue measure. We ran DPCC, DPCCE, MPCC and MPCCE for 1000 iterations. We evaluated the models using perplexity. For the two real datasets, we report perplexity on both training and test sets; for the synthetic data, we report only training perplexity. If the chain mixes well and is run sufficiently long, each sample of 5 DPCC or MPCC results used to fit the DPCCE and MPCCE models can be viewed as a sample from the DPCC or MPCC posterior distribution, respectively. We therefore also evaluated a model averaging approach, in which we calculated the perplexity based on the average of the five DPCC or MPCC likelihood results.

Table 5.2: Perplexity Comparison on Training Datasets

	Synthetic	MovieLens	Jester
LDCC	4.782 (0.025)	3.045 (0.026)	18.896 (0.072)
DPCC	3.723 (0.026)	2.797 (0.028)	15.984 (0.073)
Model Avg. of DPCC	3.687 (0.039)	2.312 (0.040)	14.223 (0.115)
DPCCE	3.573 (0.037)	2.130 (0.033)	13.677 (0.107)
MPCC	1.626 (0.023)	2.473 (0.043)	12.035 (0.088)
Model Avg. of MPCC	1.486 (0.046)	2.386 (0.051)	10.968 (0.142)
MPCCE	1.255 (0.038)	2.124 (0.037)	9.785 (0.122)

5.4.3 Evacuation Results of DPCCE and MPCCE

We present two main experimental comparisons: (a) perplexity comparisons on the synthetic data and the training sets for the real datasets; and (b) perplexity comparisons on the test sets for the real datasets.

Perplexity Comparison on Training Datasets

Figure 5.2 (left) shows the original non-grid style synthetic data matrix. After permuting its rows and columns, this matrix was input to the base co-clustering algorithms for DPCCE and MPCCE. Figure 5.2 (right) shows the corresponding grid-style partition of the original synthetic data matrix. Clearly, the grid-style partition of DPCCE over-segments the data, whereas the partition provided by MPCCE reflects the actual data distribution.

Table 5.2 shows the perplexity results for the training data. Each entry shows an average perplexity over five runs[3], with the standard deviation of the average shown in parentheses. The benefit of the non-grid partition is demonstrated by the improvement of MPCC and MPCCE over LDCC, DPCC and DPCCE. The efficacy of the ensemble approach is demonstrated by the improvement of MPCCE and DPCCE over MPCC and DPCC, respectively. The model averaging estimates perform better than their respective non-ensemble counterparts, but not as well as the ensemble estimates. All nonparametric approaches perform better than LDCC. Note that for MovieLens, MPCCE performs only 2% better than DPCCE, a difference that cannot be distinguished from sampling noise. This

[3]For DPCC and MPCC, the estimate for each run is the average of the results for the five base co-clusterings.

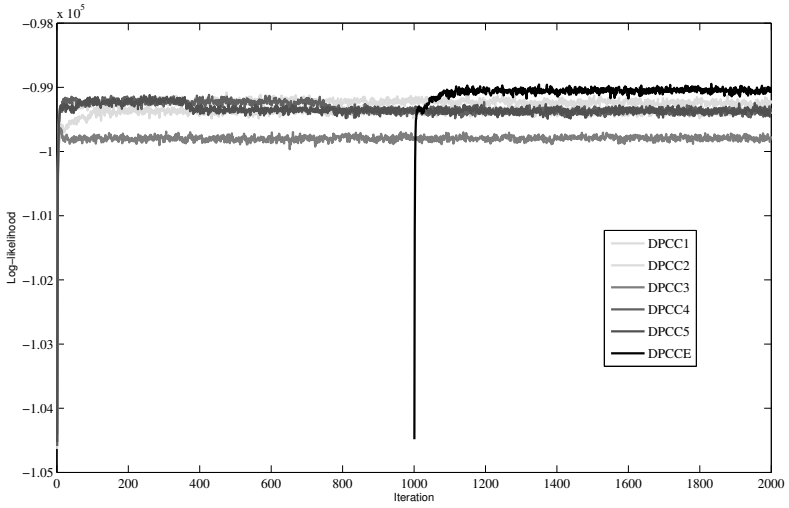


Figure 5.4: DPCC and DPCCE Likelihood Comparison

may indicate that a grid structure of independent user and movie groups provides a good fit to the MovieLens data. For the Jester dataset, the perplexities are relatively high for all models. This is due to the large number of missing values in this dataset.

All DPCCE and MPCCE experiments were run on a CentOS 5.5 server running Linux on a 4-core CPU with 4GB memory. The running time for 1000 iterations of MPCC was approximately 4 hours on MovieLens and 3 hours on Jester. For 1000 iterations of MPCCE, the running time was about 6 hours on MovieLens and 4 hours on Jester. For DPCC and DPCCE, 1000 iterations ran about 3 hours.

Figure 5.4 plots the log-likelihoods on the MovieLens dataset for 5 DPCC runs and one DPCCE run initialized with iteration 1000 of the 5 DPCC runs. Figure 5.5 plots the log-likelihoods on the Jester dataset for 5 MPCC runs and one MPCCE run initialized with iteration 1000 of the 5 MPCC runs. We also continued the DPCC and MPCC runs for another 1000 iterations to compare with DPCCE and MPCCE, respectively. All chains of DPCC and MPCC appear to have reached different local optima. The local optimum for DPCCE has higher likelihood than all five DPCC local optima, similar for MPCCE v.s. MPCC. The Potential Scale Reduction Factor MCMC diagnostic [19] for the 5 DPCC log-likelihood values plotted in Figure 5.4 is 2.9855, for the 5 MPCC log-likelihood values

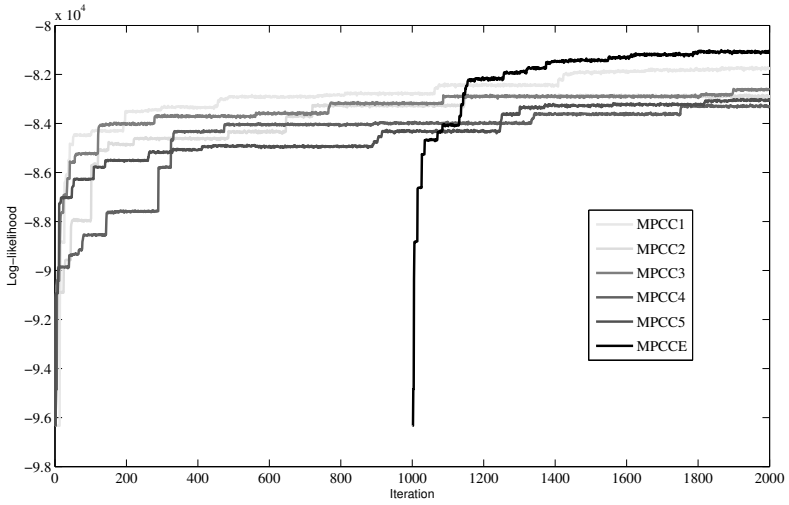


Figure 5.5: MPCC and MPCCE Likelihood Comparison

Table 5.3: Perplexity Comparison on Test Datasets

	MovieLens	Jester
LDCC	3.247 (0.052)	23.743 (0.236)
DPCC	2.908 (0.055)	20.174 (0.219)
Model Avg. of DPCC	2.838 (0.079)	19.165 (0.421)
DPCCE	2.707 (0.060)	18.092 (0.458)
MPCC	2.793 (0.067)	13.781 (0.263)
Model Avg. of MPCC	2.738 (0.089)	13.433 (0.379)
MPCCE	2.626 (0.084)	12.036 (0.438)

plotted in Figure 5.5 is 3.0043, which is also indicative of non-convergence. The other DPCC, DPCCE, MPCC and MPCCE runs followed the same pattern. These results suggest that the ensemble method finds superior local optima for samplers that mix poorly. Note running DPCCE and MPCCE for 1000 iterations requires less computation time than continuing the 5 DPCC and MPCC runs for a second 1000 iterations, and results in superior local optima.

Perplexity Comparison on Test Datasets

Predictive performance was evaluated by measuring perplexity on the test data for the two real datasets. Table 5.3 shows the prediction comparison results. Again, the results are reported as an average perplexity over multiple predictions, with the standard deviation of

each average in parentheses.

Again, all nonparametric methods perform better than LDCC; clustering ensembles perform better than model averaging, which performs better than single-run methods; and the MP methods perform better than grid-style clustering. Statistical significance tests indicate that the improvement due to the ensemble method is much greater than expected from chance variation. Mann-Whitney U-test [48] of the hypothesis that the median perplexities are the same were significant at $p < 10^{-2}$ for MPCC vs MPCCE and for DPCC vs DPCCE, on both the MovieLens and Jester data sets. Although the differences remain smaller for MovieLens than for Jester, the improvement in both MovieLens and Jester due to the non-grid partitions of the MP exceeds sampling error. That co-clustering ensembles perform better than model averaging on both training and test sets for all data sets is consistent with the hypothesis that poor mixing of the MCMC algorithms for DPCC and MPCC kept the chains near local optima of the posterior distribution, and that the ensemble algorithms can combine information from multiple local optima to find a superior co-clustering.

From Tables 5.2 and 5.3, one can observe that MPCCE doesn't improve much over DPCCE on the training and test perplexities for the MovieLens dataset, whereas MPCCE does improve significantly over DPCCE on the training and test perplexities for the Jester dataset. One possible reason is that for Jester, the kd-tree style partition fits the dataset much better than the grid-style partition. Figure 6.2 shows the co-cluster structures learned from MovieLens and Jester. One can observe that MovieLens depicts a grid-style structure, where Jester does not. Further, both the training and test perplexities of the MovieLens are much higher than those of Jester, for all models. This indicates that all models fit the MovieLens data better than the Jester data[4]. This is because Jester is a very sparse dataset (i.e., with a lot of missing values), much sparser than MovieLens, and the sparser a dataset is, the harder is to fit a model.

[4]Because perplexity is a normalized version of a likelihood, so one can directly compare perplexities of different datasets.

Chapter 6: Feature Enriched Nonparametric Bayesian Co-clustering

6.1 Introduction

Existing co-clustering techniques typically only leverage the entries of the given contingency matrix to perform the two-way clustering. As a consequence, they cannot predict the interaction values for new objects. Predictions can only be made for objects already observed (e.g., for a protein and a molecule used during training, although not necessarily in combination). This greatly limits the applicability of current co-clustering approaches.

In many applications additional features associated to the objects of interest are available, e.g., sequence information for proteins. Such features can be leveraged to perform predictions on new data. *Infinite Hidden Relational Model* (IHRM) [86] has been proposed to leverage features associated to the rows and columns of the contingency matrix to forecast relationships among previously unseen data. Although, the authors in [86] introduce IHRM from a relational learning point of view, IHRM is essentially a co-clustering model, which overcomes the aforementioned limitations of existing co-clustering techniques.

In particular, IHRM is a nonparametric Bayesian modeling of the data, that learns the number of row and column clusters from the given samples. This is achieved by adding Dirichlet Process priors on the rows and columns of the contingency matrix. As such, IHRM does not require the *a priori* specification of the numbers of row and column clusters in the data. The resulting nonparametric nature of IHRM, combined with its capability of leveraging features and making predictions for new objects, enable it to be used effectively in a large number of applications.

There are some Bayesian co-clustering models are related to IHRM, but none of them makes use of features associated to the rows and columns of the contingency matrix. A

nonparametric Bayesian co-clustering (NBCC) approach has been proposed in [49]. IHRM can be viewed as an extension of NBCC, where features associated to rows and columns are used. Such features enable IHRM to predict entries for unseen rows and columns. Many applications can greatly benefit from this prediction capability. On the contrary, existing Bayesian co-clustering models, e.g., BCC [70], LDCC [85], and NBCC [49], can handle missing entries only for already observed rows and columns (e.g., for a protein and a molecule used during training, although not necessarily in combination).

The authors in [86] have applied IHRM to collaborative filtering [66]. Although co-clustering techniques have also been applied to collaborative filtering, such as the nearest bi-clustering method [72], evolutionary co-clustering for online collaborative filtering [38] and information-theoretic co-clustering [20], none of these techniques involve features associated to rows or columns of the data matrix. IHRM, however, has the advantages of being nonparametric and of leveraging features.

While, in the original work of IHRM [86], the authors didn't explicitly evaluate how much improvement will achieve if leveraging features and making predictions for unseen objects. In this chapter, we re-interpret IHRM from the co-clustering point of view, and we rename IHRM as Feature Enriched Dirichlet Process Co-clustering (FE-DPCC). Furthermore, we focus on the empirical evaluation of forecasting relationships between previously unseen objects by leveraging object features. We conducted several experiments on a variety of relational data, including protein-molecule interaction data. The empirical evaluation demonstrates the effectiveness of the feature-enriched approach and identifies the conditions under which the use of features is most useful, i.e., with sparse data.

6.2 Feature Enriched Dirichlet Process Co-clustering

6.2.1 FE-DPCC Model

The observed data $\vec{X}$ of FE-DPCC are composed of three parts: the observed row features $\vec{X}^R$, the observed column features $\vec{X}^C$, and the observed relational features $\vec{X}^E$ between

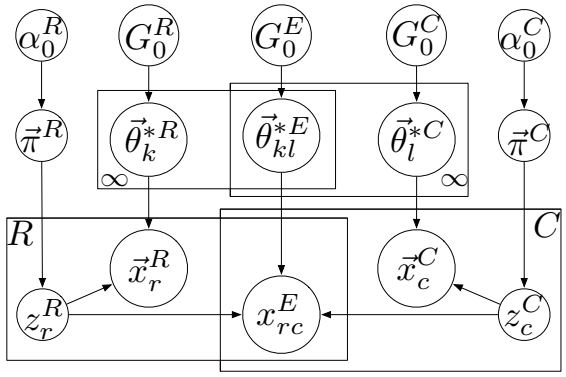


Figure 6.1: Feature Enriched Dirichlet Process Co-clustering Model

rows and columns. If there are R rows and C columns, then $\vec{X}^R = \langle \vec{x}_r^R | r = \{1, \cdots, R\} \rangle$, $\vec{X}^C = \langle \vec{x}_c^C | c = \{1, \cdots, C\} \rangle$, and $\vec{X}^E = \langle x_{rc}^E | r = \{1, \cdots, R\}, c = \{1, \cdots, C\} \rangle$. $\vec{X}^E$ may have missing data, i.e., some entries may not be observed.

FE-DPCC is a generative model and it assumes two independent DPM priors on rows and columns. We follow a stick-breaking representation to describe the FE-DPCC model. Specifically, FE-DPCC first assumes one DP prior, $\text{Dir}(\alpha_0^R, G_0^R)$, for rows, and one DP prior, $\text{Dir}(\alpha_0^C, G_0^C)$, for columns; next draws row-cluster parameters $\vec{\theta}_k^{*R}$ from G_0^R, for $k = \{1, \cdots, \infty\}$, column-cluster parameters $\vec{\theta}_l^{*C}$ from G_0^C, for $l = \{1, \cdots, \infty\}$, and co-cluster parameters $\vec{\theta}_{kl}^{*E}$ from G_0^E, for each combination of k and l[1]; then draws row mixture proportion $\vec{\pi}^R$ and column mixture proportion $\vec{\pi}^C$ as defined in Eq. 2.5. For each row r and each column c, FE-DPCC draws the row-cluster indicator z_r^R and column-cluster indicator z_c^C according to $\vec{\pi}^R$ and $\vec{\pi}^C$, respectively. Further, FE-DPCC assumes the observed features of each row r and each column c are drawn from two parametric distributions $F(\cdot|\vec{\theta}_k^{*R})$ and $F(\cdot|\vec{\theta}_l^{*C})$, respectively, and each entry, x_{rc}^E, of the relational feature matrix is drawn from a parametric distribution $F(\cdot|\vec{\theta}_{kl}^{*E})$, where $z_r^R = k$ and $z_c^C = l$.

The generative process for FE-DPCC is:

[1]Every co-cluster is indexed by a row-cluster ID and a column-cluster ID. Thus, we denote a co-cluster defined by the k^{th} row-cluster and the l^{th} column-cluster as (k, l).

- Draw $v_k^R \sim \text{Beta}(1, \alpha_0^R)$, for $k = \{1, \cdots, \infty\}$ and calculate $\vec{\pi}^R$ as in Eq (2.5)
- Draw $\vec{\theta}_k^{*R} \sim G_0^R$, for $k = \{1, \cdots, \infty\}$
- Draw $v_l^C \sim \text{Beta}(1, \alpha_0^C)$, for $l = \{1, \cdots, \infty\}$ and calculate $\vec{\pi}^C$ as in Eq (2.5)
- Draw $\vec{\theta}_l^{*C} \sim G_0^C$, for $l = \{1, \cdots, \infty\}$
- Draw $\vec{\theta}_{kl}^{*E} \sim G_0^E$, for $k = \{1, \cdots, \infty\}$ and $l = \{1, \cdots, \infty\}$
- For each row $r = \{1, \cdots, R\}$:
  - Draw $z_r^R \sim \text{Discrete}(\vec{\pi}^R)$
  - Draw $\vec{x}_r^R \sim F(\cdot|\vec{\theta}_{z_r^R}^{*R})$
- For each column $c = \{1, \cdots, C\}$:
  - Draw $z_c^C \sim \text{Discrete}(\vec{\pi}^C)$
  - Draw $\vec{x}_c^C \sim F(\cdot|\vec{\theta}_{z_c^C}^{*C})$
- For each entry $\vec{x}_{rc}^E$:
  - Draw $\vec{x}_{rc}^E \sim F(\cdot|\vec{\theta}_{z_r^R z_c^C}^{*E})$

The FE-DPCC model is illustrated in Figure 6.2.1.

6.2.2 Inference

The likelihood of the observed data is:

$$p(\vec{X}|\vec{Z}^R, \vec{Z}^C, \vec{\theta}^{*R}, \vec{\theta}^{*C}, \vec{\theta}^{*E}) = \left(\prod_{r=1}^{R} f(\vec{x}_r^R|\vec{\theta}_{z_r^R}^{*R})\right) \tag{6.1}$$

$$\left(\prod_{c=1}^{C} f(\vec{x}_c^C|\vec{\theta}_{z_c^C}^{*C})\right)\left(\prod_{r=1}^{R}\prod_{c=1}^{C} f(x_{rc}^E|\vec{\theta}_{z_r^R z_c^C}^{*E})\right),$$

where $f(\cdot|\vec{\theta}_k^{*R})$, $f(\cdot|\vec{\theta}_l^{*C})$ and $f(\cdot|\vec{\theta}_{kl}^{*E})$ denote the probability density (or mass) functions of $F(\cdot|\vec{\theta}_k^{*R})$, $F(\cdot|\vec{\theta}_l^{*C})$ and $F(\cdot|\vec{\theta}_{kl}^{*E})$, respectively; $g(\cdot|\zeta^R)$, $g(\cdot|\zeta^C)$ and $g(\cdot|\zeta^E)$ denote the probability density functions of G_0^R, G_0^C and G_0^E, respectively; $\vec{Z}^R = \langle z_r^R | r = \{1, \cdots, R\}\rangle$; $\vec{Z}^C = \langle z_c^C | c = \{1, \cdots, C\}\rangle$; $\vec{\theta}^{*R} = \langle \vec{\theta}_k^{*R} | k = \{1, \cdots, \infty\}\rangle$; $\vec{\theta}^{*C} = \langle \vec{\theta}_l^{*C} | l = \{1, \cdots, \infty\}\rangle$; and $\vec{\theta}^{*E} = \langle \vec{\theta}_{kl}^{*E} | k = \{1, \cdots, \infty\}, l = \{1, \cdots, \infty\}\rangle$.

The marginal likelihood obtained by integrating out the model parameters $\vec{\theta}^{*R}$, $\vec{\theta}^{*C}$, and $\vec{\theta}^{*E}$ is:

$$
\begin{aligned}
p(\vec{X}|\vec{Z}^R, \vec{Z}^C, G_0^R, G_0^C, G_0^E) = \\
&\left(\prod_{r=1}^{R} \int f(\vec{x}_r^R|\vec{\theta}_{z_r^R}^{*R}) g(\vec{\theta}_{z_r^R}^{*R}|\zeta^R) d\vec{\theta}_{z_r^R}^{*R}\right) \\
&\left(\prod_{c=1}^{C} \int f(\vec{x}_c^C|\vec{\theta}_{z_c^C}^{*C}) g(\vec{\theta}_{z_c^C}^{*C}|\zeta^C) d\vec{\theta}_{z_c^C}^{*C}\right) \\
&\left(\prod_{r=1}^{R}\prod_{c=1}^{C} \int f(x_{rc}^E|\vec{\theta}_{z_r^R z_c^C}^{*E}) g(\vec{\theta}_{z_r^R z_c^C}^{*E}|\zeta^E) d\vec{\theta}_{z_r^R z_c^C}^{*E}\right)
\end{aligned}
\tag{6.2}
$$

We assume $F(\cdot|\vec{\theta}_k^{*R})$ and G_0^R, $F(\cdot|\vec{\theta}_l^{*C})$ and G_0^C, and $F(\cdot|\vec{\theta}_{kl}^{*E})$ and G_0^E are all pairwise conjugate. Thus, there is a closed form expression for the marginal likelihood (6.2).

The conditional distribution for sampling the row-cluster indicator variable z_r^R for the r^{th} row $\vec{x}_r^R$ is as follows. For populated row-clusters $k \in \{Z_{r'}^R\}_{r'=\{1,\cdots,r-1,r+1,\cdots,R\}}$,

$$
\begin{aligned}
p(z_r^R = k|\vec{x}_r^R, \{x_{rc}^E\}_{c\in\{1,\cdots,C\}}, \vec{X}^{R\neg r}, \vec{X}^{E\neg r}, \vec{Z}^{R\neg r}) \\
&\propto \frac{\mathcal{N}_k^{\neg r}}{R - 1 + \alpha_0^R} \int f(\vec{x}_r^R|\vec{\theta}_k^{*R}) g(\vec{\theta}_k^{*R}|\zeta_k^{*R\neg r}) d\vec{\theta}_k^{*R} \\
&\times \prod_{c=1}^{C} \left(\int f(x_{rc}^E|\vec{\theta}_{kz_c^C}^{*E}) g(\vec{\theta}_{kz_c^C}^{*E}|\zeta_{kz_c^C}^{*E\neg r}) d\vec{\theta}_{kz_c^C}^{*E}\right)
\end{aligned}
\tag{6.3}
$$

where $\neg r$ means excluding the r^{th} row, $\mathcal{N}_k^{\neg r}$ is the number of rows assigned to the k^{th} row-cluster excluding the r^{th} row, $\zeta_k^{*R\neg r}$ is the hyperparameter of the posterior distribution of the k^{th} row-cluster parameter $\vec{\theta}_k^{*R}$ given all rows assigned to the k^{th} row-cluster excluding the r^{th} row, and $\zeta_{kz_c^C}^{*E\neg r}$ is the hyperparameter of the posterior distribution of the co-cluster (k, z_c^C) given all entries assigned to it excluding the entries in the r^{th} row. When $k \notin \{z_{r'}^R\}_{r'=\{1,\cdots,r-1,r+1,\cdots,R\}}$, i.e., z_r^R is being set to its own singleton row-cluster, the conditional distribution becomes:

$$
\begin{aligned}
& p(z_r^R = k | \vec{x}_r^R, \{x_{rc}^E\}_{c\in\{1,\cdots,C\}}, \vec{X}^{R\neg r}, \vec{X}^{E\neg r}, \vec{Z}^{R\neg r}) \qquad (6.4)\\
\propto\ & \frac{\alpha_0^R}{R-1+\alpha_0^R} \int f(\vec{x}_r^R|\vec{\theta}_k^{*R}) g(\vec{\theta}_k^{*R}|\zeta^R) d\vec{\theta}_k^{*R} \\
\times\ & \prod_{c=1}^{C} \left(\int f(x_{rc}^E|\vec{\theta}_{kz_c^C}^{*E}) g(\vec{\theta}_{kz_c^C}^{*E}|\zeta_{kz_c^C}^{*E\neg r}) d\vec{\theta}_{kz_c^C}^{*E} \right)
\end{aligned}
$$

The conditional distribution for sampling the column-cluster indicator variable z_c^C for the c^{th} column $\vec{x}_c^C$ is obtained analogously. For populated row-clusters $l \in \{Z_{c'}^C\}_{c'=\{1,\cdots,c-1,c+1,\cdots,C\}}$,

$$
\begin{aligned}
& p(z_c^C = l | \vec{x}_c^C, \{x_{rc}^E\}_{r\in\{1,\cdots,R\}}, \vec{X}^{C\neg c}, \vec{X}^{E\neg c}, \vec{Z}^{C\neg c}) \qquad (6.5)\\
\propto\ & \frac{\mathcal{N}_l^{\neg c}}{C-1+\alpha_0^C} \int f(\vec{x}_c^C|\vec{\theta}_l^{*C}) g(\vec{\theta}_l^{*C}|\zeta_l^{*C\neg c}) d\vec{\theta}_l^{*C} \\
\times\ & \prod_{r=1}^{R} \left(\int f(x_{rc}^E|\vec{\theta}_{z_r^R l}^{*E}) g(\vec{\theta}_{z_r^R l}^{*E}|\zeta_{z_r^R l}^{*E\neg c}) d\vec{\theta}_{z_r^R l}^{*E} \right)
\end{aligned}
$$

where $\neg c$ means excluding the c^{th} column, $\mathcal{N}_l^{\neg r}$ is the number of columns assigned to the l^{th} column-cluster excluding the l^{th} column, $\zeta_l^{*C\neg c}$ is the hyperparameter of the posterior distribution of the l^{th} column-cluster parameter $\vec{\theta}_l^{*C}$ given all columns assigned to the l^{th}

column-cluster excluding the c^{th} column, and $\zeta^{*E\neg c}_{z^R_r l}$ is the hyperparameter of the posterior distribution of the co-cluster (z^R_r, l) given all entries assigned to it excluding the entries in the c^{th} column. If $z^C_c \notin \{z^C_{c'}\}_{c'=\{1,\cdots,c-1,c+1,\cdots,C\}}$, i.e., z^C_c is being assigned to its own singleton column-cluster, the conditional distribution becomes:

$$
\begin{aligned}
& p(z^C_c = l | \vec{x}^C_c, \{x^E_{rc}\}_{r\in\{1,\cdots,R\}}, \vec{X}^{C\neg c}, \vec{X}^{E\neg r}, \vec{Z}^{C\neg c}) \qquad (6.6)\\
\propto\ & \frac{\alpha^C_0}{C-1+\alpha^C_0} \int f(\vec{x}^C_c | \vec{\theta}^{*C}_l) g(\vec{\theta}^{*C}_l | \zeta^C) d\vec{\theta}^{*C}_l \\
\times\ & \prod_{r=1}^{R} \left(\int f(x^E_{rc} | \vec{\theta}^{*E}_{z^R_r l}) g(\vec{\theta}^{*E}_{z^R_r l} | \zeta^{*E\neg c}_{z^R_r l}) d\vec{\theta}^{*E}_{z^R_r l} \right)
\end{aligned}
$$

Table 6.1 summarizes the notation used in this section.

6.3 Experimental Evaluation

6.3.1 Datasets

We conducted experiments on two rating datasets and two protein-molecule interaction datasets. MovieLens[2] is a movie recommendation dataset containing 100,000 ratings in a sparse data matrix for 1682 movies rated by 943 users. Jester[3] is a joke rating dataset. The original dataset contains 4.1 million continuous ratings of 140 jokes from 73,421 users. We chose a subset containing 100,000 ratings. Following [70], we uniformly discretized the ratings into 10 bins. The protein-molecule interaction datasets are described in Section 6.3.2 below. Table 6.2 summarizes the characteristics of the four datasets.

[2]http://www.grouplens.org/node/73

[3]http://goldberg.berkeley.edu/jester-data/

6.3.2 Protein-Molecule Interaction Study

Small organic molecules (a.k.a. ligands) can bind to different proteins and modulate (inhibit/activate) their functions. Understanding these interactions provides insight into the underlying biological processes and is useful for designing therapeutic drugs [67, 76]. Small molecules can work rapidly and reversibly, can modulate a single function of a multifunction protein, and can disrupt protein-protein interactions. In this work we use the FE-DPCC approach to co-cluster the relational data obtained from different protein-molecule interaction studies along with standard features extracted for a protein target and the chemical molecules.

The first protein-molecule interaction dataset (MP1[4]) consists of G-protein coupled receptor (GPCR) proteins and their interaction with small molecules [32]. GPCRs are used widely in the pharmaceutical industry as a therapeutic target. We used sequence features and hierarchical features of proteins for the MP1 dataset. When using protein sequence features, we extracted k-mer features. These interactions are the product of an assay (biological experiment) that evaluates whether a particular protein target is active against a molecule. In our dataset MP1, we had 4051 interactions between 166 proteins and 2687 molecules. The use of targets restricted to a specific group of proteins (GPCRs) is similar to a *chemogenomics* approach where the assumption is that proteins belonging to the same family have a similarity in their interaction or activity profile.

We also evaluated our algorithm on an additional protein-molecule interaction dataset (MP2[5]), used previously in [57]. This dataset is different from MP1, in the sense that the protein targets belong to a more general class and are not restricted to GPCRs. We used protein sequence features and extracted 5-mer features. In this dataset we had 154 proteins, 45408 molecules, and a total of 154×45408 interactions. MP2 is very sparse; we therefore selected the subset of molecules that interact with at least two proteins, resulting in 2876 molecules and a total of 7146 positive interactions.

[4] http://pharminfo.pharm.kyoto-u.ac.jp/services/glida/
[5] http://pubchem.ncbi.nlm.nih.gov/

6.3.3 Methodology

For fair comparison, we compared FE-DPCC with a variant of NBCC, called *Dirichlet Process Co-clustering* (DPCC), which assumes two independent Dirichlet Process priors on rows and columns, as FE-DPCC does. FE-DPCC uses row and column features, whereas DPCC does not. In the original work of NBCC [49], the authors used Pitman-Yor Process priors, a generalization of Dirichlet Processes. We ran 1000 iterations of Gibbs sampling for both FE-DPCC and DPCC.

We used perplexity as an evaluation metric on the test data. The perplexity of a dataset D is defined as $perplexity(D) = \exp\left(-\frac{\mathcal{L}(D)}{N}\right)$, where $\mathcal{L}(D)$ is the log-likelihood of D, and N is the number of data points in D. The higher the log-likelihood, the lower the perplexity, and the better a model fits the data. For models that provide probabilistic predictions of test data, perplexity is a better metric than accuracy, because it takes into account the model's confidence in its prediction – assigning greater penalty when the model is more certain of its erroneous response.

The relational features in our data are discrete. We assume $f(\cdot|\vec{\theta}^{*E}_{kl})$ is a categorical distribution, denoted as $\text{Cat}(\cdot|\vec{\theta}^{*E}_{kl})$, and $g(\vec{\theta}^{*E}_{kl}|\zeta^E)$ is a Dirichlet distribution, denoted as $\text{Dir}(\vec{\theta}^{*E}_{kl}|\vec{\varphi})$, with $\zeta^E = \vec{\varphi}$. Because of conjugacy, we can marginalize out $\vec{\theta}^{*E}_{kl}$. Without loss of generality, we assume that $f(\cdot|\vec{\theta}^{*E}_{kl})$ is a D-dimensional categorical distribution with support $\{1, \cdots, D\}$, and we denote the Dirichlet hyperparameter as $\zeta^E = \vec{\varphi} = \langle \varphi_d | d = \{1, \cdots, D\} \rangle$. The predictive distribution of the co-cluster (k, l) observing a new entry $x^E_{r'c'} = d$, $d \in \{1, \cdots, D\}$, is:

$$
\begin{aligned}
p(x^E_{r'c'} = d | \zeta^{*E}_{kl}, z^R_{r'} = k, z^C_{c'} = l) = \\
\int f(x^E_{r'c'} = d | \vec{\theta}^{*E}_{kl}) g(\vec{\theta}^{*E}_{kl} | \zeta^{*E}_{kl}) d\vec{\theta}^{*E}_{kl}) = \\
\int \text{Cat}(x^E_{r'c'} = d | \vec{\theta}^{*E}_{kl}) \text{Dir}(\vec{\theta}^{*E}_{kl} | \vec{\varphi}^{*}_{kl}) d\vec{\theta}^{*E}_{kl}) \propto \mathcal{N}^d_{(k,l)} + \varphi_d
\end{aligned}
\tag{6.7}
$$

where $\vec{\varphi}^*_{kl}$ is the posterior hyperparameter of the Dirichlet distribution of the co-cluster (k, l), and $\mathcal{N}^d_{(k,l)}$ is the number of entries assigned to the co-cluster (k, l) and equal to d.

In the MovieLens dataset, rows represent users and columns represent movies. Row features are age, gender, and occupation; column features form a 19-dimensional binary vector where a non-zero dimension means the movie belongs to the corresponding genre, for a total of 19 genres. We assumed independence among the row features and the column features conditional on row- and column-clusters. We modeled age as drawn from a Poisson distribution, $\text{Poi}(\cdot|\lambda)$, with a conjugate Gamma prior, $\text{Gamma}(\lambda|\varrho, \varsigma)$. We modeled gender as drawn from a Bernoulli distribution, $\text{Ber}(\cdot|\vartheta)$, with a conjugate Beta prior $\text{Beta}(\vartheta|\varkappa, \varpi)$. The occupation feature is categorical, modeled as $\text{Cat}(\cdot|\vec{\phi})$, with Dirichlet prior, $\text{Dir}(\vec{\phi}|\vec{\varphi})$. Thus, the row feature parameter is given by $\vec{\theta}^{*R}_k = \langle \lambda^*_k, \vartheta^*_k, \vec{\phi}^*_k \rangle$, and the row feature prior hyperparameter is $\zeta^R = \langle \varrho, \varsigma, \vartheta, \vec{\varphi} \rangle$. We denote the feature vector of a new user as $\vec{x}^R_{r'} = \langle a_{r'}, g_{r'}, o_{r'} \rangle$, where $a_{r'}$, $g_{r'}$, and $o_{r'}$ represent the age, gender and occupation, respectively. The predictive distribution of the k^{th} row-cluster observing a new user, $\vec{x}^R_{r'}$, is:

$$
\begin{aligned}
p(\vec{x}^R_{r'}|\varrho^*_k, \varsigma^*_k, \varkappa^*_k, \varpi^*_k, \vec{\varphi}^*_k, z^R_{r'} = k) = \\
\left(\int \text{Poi}(a_{r'}|\lambda^*_k) \text{Gamma}(\lambda^*_k|\varrho^*_k, \varsigma^*_k) d\lambda^*_k \right) \\
\left(\int \text{Ber}(g_{r'}|\vartheta^*_k) \text{Beta}(\vartheta^*_k|\varkappa^*_k, \varpi^*_k) d\vartheta^*_k \right) \\
\left(\int \text{Cat}(o_{r'}|\vec{\phi}^*_k) \text{Dir}(\vec{\phi}^*_k|\vec{\varphi}^*_k) d\vec{\phi}^*_k \right)
\end{aligned}
\tag{6.8}
$$

where ϱ^*_k, ς^*_k, $\varkappa^*_k$, ϖ^*_k, and $\vec{\varphi}^*_k$ are the posterior hyperparameters (k indices the row-clusters). Denote $\zeta^{*R}_k = \langle \varrho^*_k, \varsigma^*_k, \varkappa^*_k, \varpi^*_k, \vec{\varphi}^*_k \rangle$. We also assume that features associated to movies (columns) are generated from a Multinomial distribution, $\text{Mul}(\cdot|\vec{\psi})$, with Dirichlet prior, $\text{Dir}(\vec{\psi}|\vec{\varphi})$. Accordingly, $\vec{\theta}^{*C}_l = \vec{\psi}^*_l$, and $\zeta^C = \vec{\varphi}$. The predictive distribution of the l^{th}

column-cluster observing a new movie, $\vec{x}^C_{c'}$, is:

$$p(\vec{x}^C_{c'}|\vec{\varphi}^*_l, z^C_{c'} = l) = \int \text{Mul}(\vec{x}^C_{c'}|\vec{\psi}^*_l)\text{Dir}(\vec{\psi}^*_l|\vec{\varphi}^*_l)d\vec{\psi}^*_l$$

where $\zeta^{*C}_l = \vec{\varphi}^*_l$ is the posterior hyperparameter of the Dirichlet distribution (l indices the column-clusters).

In the Jester dataset, rows represent users and columns represent jokes. No features are associated to users, thus row-clusters cannot predict an unseen user. We used a bag-of-word representation for joke features, and assumed each joke feature vector is generated from a Multinomial distribution, $\text{Mul}(\cdot|\vec{\psi})$, with a Dirichlet prior, $\text{Dir}(\vec{\psi}|\vec{\varphi})$. The predictive distribution of the l^{th} column-cluster observing a new joke, $\vec{x}^C_{c'}$, is:

$$p(\vec{x}^C_{c'}|\vec{\varphi}^*_l, z^C_{c'} = l) = \int \text{Mul}(\vec{x}^C_{c'}|\vec{\psi}^*_l)\text{Dir}(\vec{\psi}^*_l|\vec{\varphi}^*_l)d\vec{\psi}^*_l$$

For the two protein-molecule interaction datasets, rows represent molecules and columns represent proteins. We extracted k-mer features from protein sequences. For MP1, we also used hierarchical features for proteins. We used a graph-fragment-based feature representation that computes the frequency of different length cycles and paths for each molecule. These graph-fragment-based features were derived using a chemoinformatics toolkit called AFGEN [82] (default parameters were used) and are known to capture structural aspects of molecules effectively. In both cases, we assumed each protein is generated from a Multinomial distribution, $\text{Mul}(\cdot|\vec{\psi}^p)$, with a Dirichlet prior, $\text{Dir}(\vec{\psi}^p|\vec{\varphi}^p)$. We also assumed each molecule is generated from a Multinomial distribution, $\text{Mul}(\cdot|\vec{\psi}^m)$, with a Dirichlet prior, $\text{Dir}(\vec{\psi}^m|\vec{\varphi}^m)$. The predictive distribution of the k^{th} row-cluster observing a new molecule, $\vec{x}^R_{r'}$, is:

$$p(\vec{x}^R_{r'}|\vec{\varphi}^{*m}_k, z^R_{r'} = k) = \int \text{Mul}(\vec{x}^R_{r'}|\vec{\psi}^{*m}_k)\text{Dir}(\vec{\psi}^{*m}_k|\vec{\varphi}^{*m}_k)d\vec{\psi}^{*m}_k$$

The predictive distribution of the l^{th} column-cluster observing a new protein, $\vec{x}^C_{c'}$, is:

$$p(\vec{x}^C_{c'}|\vec{\varphi}^{*p}_l, z^C_{c'} = l) = \int \text{Mul}(\vec{x}^C_{c'}|\vec{\psi}^{*p}_l)\text{Dir}(\vec{\psi}^{*p}_l|\vec{\varphi}^{*p}_l)d\vec{\psi}^{*p}_l$$

6.3.4 Results

We performed a series of experiments to evaluate the performance of FE-DPCC across the four datasets. All experiments were performed five times, and we report the average (and standard deviation) perplexity across the five runs. The experiments were performed on an Intel four core, Linux server with 4GB memory. The average running time for FE-DPCC was 1, 3, 3.5 and 2.5 hours on the MovieLens, Jester, MP1 and MP2 datasets, respectively.

Feature Enrichment Evaluation

Table 6.3 shows the average perplexity values (and standard deviations) across five runs for the four datasets on the test data. To analyze the effect of new rows and columns on the prediction capabilities of the algorithms, we split each test dataset into subsets based on whether the subset contains new rows or columns.

Table 6.3 shows that the overall perplexity of FE-DPCC is lower than that of DPCC on all datasets, with an improvement of 12%, 1.5%, 84% and 81% for MovieLens, Jester, MP1 and MP2, respectively. In particular, as expected, FE-DPCC is significantly better than DPCC on the portion of the test data that contains unseen rows or columns, or both. These test sets consist of entries for rows and columns that are independent and not included in the training set. The DPCC algorithm does not use features; as such it can predict entries for the new rows and columns using prior probabilities only. In contrast, the FE-DPCC algorithm leverages features along with prior probabilities; this enables our approach to predict values for the independent test entries more accurately. This ability is a major strength of our FE-DPCC algorithm. For the portion of the test data whose rows and columns are observed in the training as well, the perplexity values of FE-DPCC and DPCC

are comparable. The standard deviations indicate that the algorithms are stable, yielding consistent results across different runs.

To accurately assess the performance of the FE-DPCC algorithm, we performed a set of experiments that involved a perturbation of the protein and molecule features on the MP1 dataset. The average perplexity values on the MP1 test sets are reported in Table 6.4. k-mer= 5 has been used as protein sequence features. First, we take the protein sequences (i.e., columns) and shuffle the ordering of the amino acids. This alters the ordering of the protein sequence but maintains the same composition (i.e., the shuffled sequences have the same number of characters or amino acids). We refer to this scheme as "Shuffle". It achieves an average perplexity of 3.034, versus the average perplexity of 1.450 achieved by FE-DPCC (with no shuffling of features).

We also devised a scheme in which the row and/or column features are exchanged, e.g., the features of a particular molecule are exchanged with the features of another molecule. Such an exchange, either of proteins, molecules, or both, causes the inclusion of incorrect information within the FE-DPCC algorithm. Our aim was to assess the strength of FE-DPCC when enriched with meaningful and correct features. We refer to this scheme as "Exchange." Table 6.4 reports the results of exchanging molecule features only (Exchange M), protein features only (Exchange P), and both (Exchange M and P). We noticed an average perplexity of 2.9 in each case.

We also evaluated the FE-DPCC algorithm when only molecule or only protein features are used ("Use Only M" and "Use only P" in Table 6.4). For the DPCC algorithm (with no features), the use of only one set of features prevents the co-clustering algorithm from making inferences on the unseen rows or columns in the test set. As such, we observe a high perplexity value in Table 6.4 for these settings.

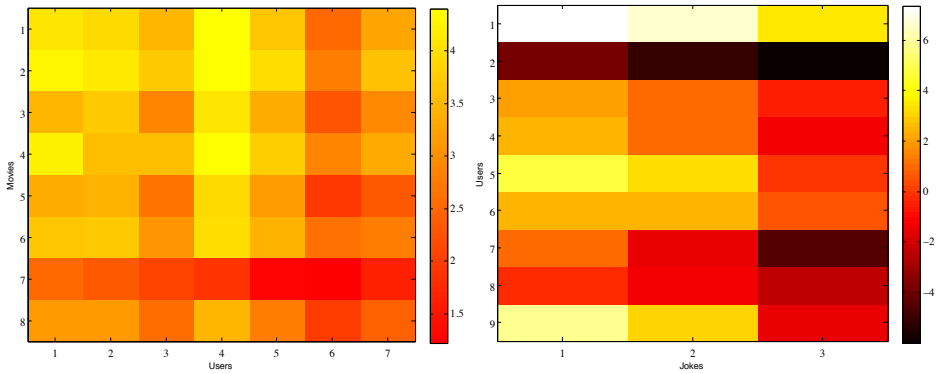
From Table 6.4 we can see that the use of incorrect features, either of proteins, or molecules, or both, hurts the prediction performance. The use of protein features only, or molecule features only, gives worse performance than the use of incorrect features. This is because, in the former case, the model observes entries of unseen proteins or unseen

molecules with low prior probabilities. The use of protein and molecule features (last row of Table 6.4) gives the best performance, which establishes the success of our technique.

For MP1 we performed an additional set of experiments to evaluate the sequence features. The k-mer features are overlapping subsequences of a fixed length extracted from the protein sequences. We used k-mer lengths of 2, 3, 4 and 5, and observed that the average perplexity (Table 6.5) remained fairly similar. As such, we used k-mer$= 5$ in all the experiments. We also compared the sequence features for the proteins to an alternate feature derived from a hierarchical biological annotation of the proteins. For the MP1 dataset the hierarchical features were extracted as done in the previous study [32, 58]. From Table 6.5 we observe that the hierarchical features (HF) achieved a slightly lower perplexity in comparison to the k-mer$= 5$ sequence features. This is encouraging, as it suggests that sequence features perform similarly to manual annotation (hierarchy), that may not be easily available across all the proteins.

Visualization of Co-clusters

In Figure 6.2 we illustrate the co-cluster structures learned by FE-DPCC on MovieLens and Jester (we do not plot the co-clusters for MP1 and MP2 because they are too sparse). We calculate the mean entry value for each co-cluster, and plot the resulting mean values in different color scales; the lighter the color is, the larger the value.



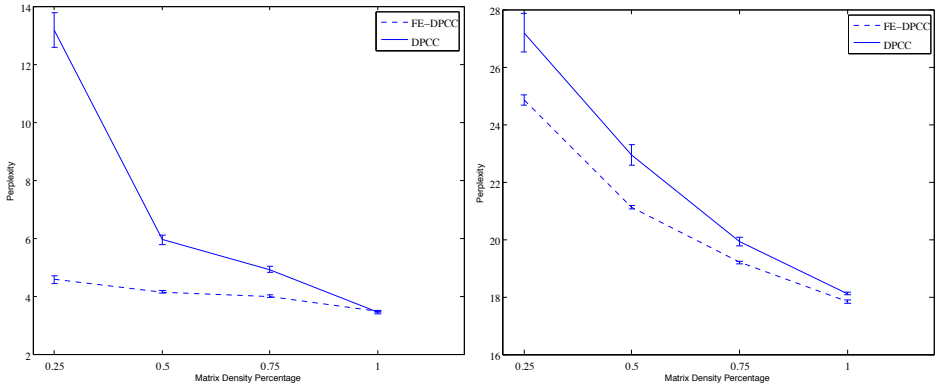
(a) MovieLens (b) Jester

Figure 6.2: Co-clusters Learned by FE-DPCC

Figure 6.2 shows that FE-DPCC was able to identify a meaningful co-cluster structure for both MovieLens and Jester.

Data Density

We also varied the density of MovieLens and Jester to see how different levels of density affect the test perplexity of FE-DPCC and DPCC. We varied the matrix density by randomly sampling 25%, 50% and 75% of the entries in the training data. These sampled matrices were then given in input to DPCC and FE-DPCC to train a model, and infer unknown entries on the test data. Figure 6.3 illustrates these results averaged across five iterations. As the sparsity of the relational matrix increases the test perplexity increases for both FE-DPCC and DPCC. But the DPCC algorithm has far higher perplexity in comparison to the FE-DPCC algorithm for a sparser matrix. As the matrix sparsity increases, the information within the relational matrix is lost and the FE-DPCC algorithm relies on the row and column features. Thus, for sparser matrices the FE-DPCC algorithm shows far better clustering results in comparison to the DPCC. These experiments suggest the reason why we see a dramatic difference between the two algorithms for the MP1 and MP2 datasets, which are very sparse (see Table 6.2). It also identifies the (sparse) conditions under which features are most useful.



(a) MovieLens (b) Jester

Figure 6.3: Test Perplexity with Different Data Densities

MCMC Convergence Diagnostics

We used 1000 iterations of Gibbs sampling for the FE-DPCC algorithm. Here we conduct MCMC convergence diagnostics to check whether 1000 iterations are enough to reach convergence. In Table 6.6, we report the potential scale reduction factor (PSRF) [19] on the log-likelihoods of five Gibbs sampling runs. PSRF compares within-chain variance with between-chain variance. Values far from 1 are diagnostic of non-convergence. Typically, values above 1.1 or 1.2 are considered problematic. From Table 6.6, we can see PSRF values on all datasets except Jester are well below 1.1, which suggests that the number of iterations is sufficient and the results have converged. The PSRF value for Jester is above 1.1 but below 1.2, suggesting possible convergence issues.

Table 6.1: Notation Description of FE-DPCC

R	Number of rows
C	Number of columns
$\vec{X}^R$	Row features, $\vec{X}^R = \langle \vec{x}_r^R \vert r = \{1, \cdots, R\} \rangle$
$\vec{X}^C$	Column features, $\vec{X}^C = \langle \vec{x}_c^C \vert c = \{1, \cdots, C\} \rangle$
$\vec{X}^E$	Relational features between rows and columns, $\vec{X}^E = \langle x_{rc}^E \vert r = \{1, \cdots, R\}, c = \{1, \cdots, C\} \rangle$
k	Index for row-clusters
l	Index for column-clusters
z_r^R	Row-cluster indicator variable for the r^{th} row, and $z_r^R \in \{1, \cdots, k, \cdots, \infty\}$
z_c^C	Column-cluster indicator variable for the c^{th} column, and $z_c^C \in \{1, \cdots, l, \cdots, \infty\}$
$\vec{Z}^R$	$\vec{Z}^R = \langle z_r^R \vert r \in \{1, \cdots, R\} \rangle$
$\vec{Z}^C$	$\vec{Z}^C = \langle z_c^C \vert c \in \{1, \cdots, C\} \rangle$
$\vec{\theta}_k^{*R}$	Parameter of the k^{th} row-cluster
$\vec{\theta}_l^{*C}$	Parameter of the l^{th} column-cluster
$\vec{\theta}_{kl}^{*E}$	Parameter of the co-cluster (k, l)
ζ^R	Hyperparameter of the prior distribution to the k^{th} row-cluster parameter
ζ^C	Hyperparameter of the prior distribution to the l^{th} column-cluster parameter
ζ^E	Hyperparameter of the prior distribution to the parameter of the co-cluster (k, l)
$\vec{Z}^{R \neg r}$	$\vec{Z}^{R \neg r} = \langle z_{r'}^R \vert r' \in \{1, \cdots, R\}, r' \neq r \rangle$
$\vec{X}^{E \neg r}$	$\vec{X}^{E \neg r} = \langle x_{r'c}^E \vert r' = \{1, \cdots, R\}, r' \neq r, c = \{1, \cdots, C\} \rangle$
$\mathcal{N}_k^{\neg r}$	Number of rows assigned to the k^{th} row-cluster, excluding the r^{th} row
$\vec{Z}^{C \neg c}$	$\vec{Z}^{C \neg c} = \langle z_{c'}^C \vert c' \in \{1, \cdots, C\}, c' \neq c \rangle$
$\vec{X}^{E \neg c}$	$\vec{X}^{E \neg c} = \langle x_{rc'}^E \vert r = \{1, \cdots, R\}, c' = \{1, \cdots, C\}, c' \neq c \rangle$
$\mathcal{N}_l^{\neg c}$	Number of columns assigned to the l^{th} column-cluster, excluding the c^{th} column
$g(\vec{\theta}_k^{*R} \vert \zeta_k^{*R})$	Posterior distribution of the parameter of the k^{th} row-cluster
$g(\vec{\theta}_k^{*R} \vert \zeta_k^{*R \neg r})$	Posterior distribution of the parameter of the k^{th} row-cluster, updated without the r^{th} row
$\zeta_k^{*R \neg r}$	Hyperparameter of the posterior distribution of the parameter of the k^{th} row-cluster, updated without the r^{th} row
$g(\vec{\theta}_{kl}^{*E} \vert \zeta_{kl}^{*E})$	Posterior distribution of the parameter of the co-cluster (k, l)
$g(\vec{\theta}_{kl}^{*E} \vert \zeta_{kl}^{*E \neg r})$	Posterior distribution of the parameter of the co-cluster (k, l), updated without the entries in the r^{th} row
$\zeta_{kl}^{*E \neg r}$	Parameter of the posterior distribution of the parameter of the co-cluster (k, l), updated without the entries in the r^{th} row
$g(\vec{\theta}_l^{*C} \vert \zeta_l^{*C})$	Posterior distribution of the parameter of the l^{th} column-cluster
$g(\vec{\theta}_l^{*C} \vert \zeta_l^{*C \neg c})$	Posterior distribution of the parameter of the l^{th} column-cluster, updated without the c^{th} column
$\zeta_l^{*C \neg c}$	Hyperparameter of the posterior distribution of the parameter of the l^{th} column-cluster, updated without the c^{th} column
$g(\vec{\theta}_{kl}^{*E} \vert \zeta_{kl}^{*E \neg c})$	Posterior distribution of the parameter of the co-cluster (k, l), updated without the entries in the c^{th} column
$\zeta_{kl}^{*E \neg c}$	Hyperparameter of the posterior distribution of the parameter of the co-cluster (k, l), updated without the entries in the c^{th} column

Table 6.2: Training and Test Data of Each Dataset

		MovieLens	Jester	MP1	MP2
Train	# Rows	943	33459	1961	2674
	# Columns	1650	140	61	149
	# Entries	80000	80000	3000	5000
	Density	5.142%	1.708%	2.508%	1.255%
Test	# Rows	927	14523	856	1647
	# Columns	1407	139	68	145
	# Entries	20000	20000	1051	2146
	Density	1.533%	0.991%	1.806%	0.899%

Table 6.3: Test Perplexity of Different Test Subsets

		MovieLens	Jester	MP1	MP2
DPCC	Row and Column Observed	3.327 (0.020)	17.111 (0.031)	1.430 (0.011)	1.484 (0.013)
	Row or Column Unseen	4.427 (0.047)	19.322 (0.025)	8.845 (0.011)	7.987 (0.011)
	Overall Perplexity	4.424 (0.087)	18.116 (0.035)	8.843 (0.013)	7.980 (0.021)
FE-DPCC	Row and Column Observed	3.344 (0.021)	17.125 (0.040)	1.435 (0.024)	1.489 (0.023)
	Row or Column Unseen	3.892 (0.026)	17.836 (0.053)	1.453 (0.026)	1.509 (0.024)
	Overall Perplexity	3.889 (0.031)	17.836 (0.062)	1.450 (0.046)	1.501 (0.045)

Table 6.4: Test Perplexity of (M)olecule and (P)rotein Features on the MP1 Dataset

	Perplexity
Shuffle P	3.034 (0.083)
Exchange M	2.945 (0.083)
Exchange P	2.932 (0.071)
Exchange M and P	2.991 (0.095)
Use Only M	7.235 (0.043)
Use Only P	7.789 (0.045)
Use M and P	1.450 (0.046)

Table 6.5: Test Perplexity of Protein Features of the MP1 Dataset

	Perplexity
2-mer	1.471 (0.057)
3-mer	1.437 (0.044)
4-mer	1.441 (0.049)
5-mer	1.450 (0.046)
HF	1.413 (0.010)

Table 6.6: MCMC Diagnostics

Dataset	PSRF
MovieLens	1.023
Jester	1.186
MP1 (HF)	1.038
MP1 (SF)	1.046
MP2 (SF)	1.053

Chapter 7: Conclusion and Future Work

7.1 Conclusion

In conclusion, clustering is an important unsupervised learning problem that arises in a variety of applications for data analysis and mining. However, clustering is an ill-posed problem and, as such, a challenging one: no ground-truth that can be used to validate clustering results is available. Two issues arise as a consequence. Various clustering algorithms embed their own bias resulting from different optimization criteria. As a result, each algorithm may discover different patterns in a given dataset. The second issue concerns the setting of parameters. In clustering, parameter setting controls the characterization of individual clusters, and the total number of clusters in the data. In addition, the high-dimensionality of the data, which is commonly seen in practice, makes the clustering process even more difficult.

There has been some existing work to address the issues of clustering. Clustering ensembles have been proposed to address the issue of different biases induced by various algorithms. Bayesian and nonparametric Bayesian approaches have been applied to clustering to address the parameter tuning and model selection issues. Subspace clustering, or co-clustering, is proposed to address the dimensionality issue of clustering.

Although attempts have been made in the literature to address individually the major issues related to clustering, no previous work has addressed them jointly. In my dissertation, I introduced a unified framework that addresses all three issues at the same time.

7.2 Contributions

Specifically, I designed a nonparametric Bayesian clustering ensemble (NBCE) approach, which assumes that multiple observed clustering results are generated from an unknown

consensus clustering. The underlying distribution is assumed to be a mixture distribution with a nonparametric Bayesian prior. The number of mixture components, i.e., the number of consensus clusters, is learned automatically. By combining the ensemble methodology and nonparametric Bayesian modeling, NBCE addresses both the ill-posed nature and the parameter setting/model selection issues of clustering. Furthermore, NBCE outperforms individual clustering methods, since it can escape local optima by combining multiple clustering results. I also designed a nonparametric Bayesian co-clustering ensemble (NBCCE) technique. NBCCE inherits the advantages of NBCE, and in addition it is effective with high dimensional data. As such, NBCCE provides a unified framework to address all the three aforementioned issues.

Further, I evaluated a novel feature-enriched nonparametric Bayesian Co-clustering approach, which can predict relationships between previously unseen objects and is most effective with sparse data. Large improvements have been achieved with protein-molecule interaction data.

7.3 Future Work

As for the future work, my work can be extended in several ways. First, Bayesian methods are computationally very intensive. How to speed up Bayesian inference is an open issue in Bayesian research. Recently, GPUs have drawn a lot of attention in parallel computing. A GPU is a highly parallel, multithreaded, and multi-core processor. Traditionally, GPUs were designed for graphics computation, but more recently they have been used for general parallel computation. GPUs can possibly be used to also speed up Bayesian inference.

Second, FE-DPCC still assumes independence between row-clusters and column-clusters. It is possible to replace the two independent Dirichlet Process priors in FE-DPCC with a Mondrian Process prior. The resulting model will provide more flexibility in fitting the data.

Third, I have noticed that the inference for Mondrian Processes is very complex and time consuming. The MCMC inference for MP involves RJMCMC to draw Mondrian samples and Gibbs sampling to draw indicator variables to assign data to subspace clusters.

Mondrian samples are used to define kd-tree (subspace cluster) structures, which are structure dependent terms, and the indicator variables are used for data assignments, which are data dependent terms. There are two separate inference stages for MP, one for the inference of the structure terms, the other for the inference of the data terms. I have been thinking of extending Dirichlet Diffusion Trees (DDT) [55, 56] to handle co-clustering. DDT is a nonparametric Bayesian model for hierarchical clustering. DDT assumes a prior distribution over binary trees. As such, both MP and DDT are tree structure priors. One advantage of DDT inference is that it is possible to marginalize out the data dependent terms of DDT, and only focus on the inference of structure terms, which makes DDT inference more efficient.

Bibliography

Bibliography

[1] D. G. Alexey, A. Tsymbal, N. Bolshakova, and P. Cunningham. Ensemble clustering in medical diagnostics. In *IEEE Symposium on Computer-Based Medical Systems*, pages 576–581. IEEE Computer Society, 2004.

[2] C. E. Antoniak. Mixtures of Dirichlet Processes with Applications to Bayesian Nonparametric Problems. *The Annals of Statistics*, 2(6):1152–1174, 1974.

[3] H. Ayad and M. Kamel. Finding natural clusters using multi-clusterer combiner based on shared nearest neighbors. In *Fourth International Workshop on Multiple Classifier Systems*, volume 2709, pages 166–175. Springer, 2003.

[4] M. J. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

[5] D. Blackwell and J. B. Macqueen. Ferguson distributions via Pólya urn schemes. *The Annals of Statistics*, 1:353–355, 1973.

[6] D. M. Blei, T. L. Griffiths, and M. I. Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 57(2):1–30, 2010.

[7] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical Topic Models and the Nested Chinese Restaurant Process. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

[8] D. M. Blei and J. D. Mcauliffe. Supervised topic models. In *Advances in Neural Information Processing Systems (NIPS)*, volume 21, 2007.

[9] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, 2003.

[10] Y. Cheng and G. M. Church. Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103. AAAI Press, 2000.

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

[12] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM, 2003.

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

[14] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

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

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

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

[18] A. L. N. Fred and A. K. Jain. Data clustering using evidence accumulation. In *International Conference on Pattern Recognition*, volume 4, pages 276–280, Washington, DC, USA, 2002. IEEE Computer Society.

[19] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis, Second Edition (Texts in Statistical Science)*, chapter 1, 2, 3, 5, 9 and 11. Chapman & Hall/CRC, 2 edition, July 2003.

[20] T. George and S. Merugu. A scalable collaborative filtering framework based on co-clustering. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, ICDM '05, pages 625–628, Washington, DC, USA, 2005. IEEE Computer Society.

[21] D. Gondek and T. Hofmann. Non-redundant clustering with conditional ensembles. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 70–77. ACM, 2005.

[22] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[23] P. J. Green. Trans-dimensional markov chain monte carlo. In P. J. Green, N. L. Hjort, and S. T. Richardson, editors, *Highly Structured Stochastic Systems*, pages 179–198. Oxford University Press, Oxford, UK, 2003.

[24] T. L. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. In *Advances in Neural Information Processing Systems (NIPS)*, volume 18, 2006.

[25] T. L. Griffiths and M. Steyvers. Finding scientific topics. In *Proceedings of the National Academy of Sciences*, volume 101, pages 5228–5235, 2004.

[26] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.

[27] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2 edition, July 2003.

[28] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, April 1970.

[29] F. Hoppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition*. Wiley, 1999.

[30] X. Hu. Integration of cluster ensemble and text summarization for gene expression analysis. In *Fourth IEEE Symposium on Bioinformatics and Bioengineering*, pages 251–258, 2004.

[31] H. Ishwaran and L. James. Gibbs sampling methods for stick breaking priors. *Journal of the American Statistical Association*, 96(453):161–173, March 2001.

[32] L. Jacob, B. Hoffmann, V. Stoven, and J.-P. Vert. Virtual screening of GPCRs: an in silico chemogenomics approach. *BMC Bioinformatics*, 9(1):363, 2008.

[33] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[34] S. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, Sept. 1967.

[35] M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

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

[37] C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, 2006.

[38] M. Khoshneshin and W. N. Street. Incremental collaborative filtering via evolutionary co-clustering. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 325–328, New York, NY, USA, 2010. ACM.

[39] L. I. Kuncheva. Experimental comparison of cluster ensemble methods. In *International Conference on Information Fusion*, pages 1–7, 2006.

[40] L. I. Kuncheva and S. T. Hadjitodorov. Using diversity in cluster ensembles. In *International Conference on Systems, Man and Cybernetics*, volume 2, pages 1214–1219, 2004.

[41] L. I. Kuncheva and D. Vetrov. Evaluation of stability of k-means cluster ensembles with respect to random initializationtent semantic analysis. *PAMI*, 28(11):1798–1808, 2006.

[42] K. Kurihara, M. Welling, and Y. W. Teh. Collapsed variational dirichlet process mixture models. In *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pages 2796–2801, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[43] W. Li, D. Blei, and A. Mccallum. Nonparametric bayes pachinko allocation. In *UAI 07*, 2007.

[44] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 577–584. ACM, 2006.

[45] S. P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28:129–137, 1982.

[46] D. J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2002.

[47] J. B. Macqueen. Some methods for classification and analysis of multivariate observations. In *Procedings of the Fifth Berkeley Symposium on Math, Statistics, and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[48] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18(1):50–60, 1947.

[49] E. Meeds and S. Roweis. Nonparametric Bayesian Biclustering. Technical Report UTML TR 2007-001, Department of Computer Science, University of Toronto, 2007.

[50] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, June 1953.

[51] D. Mimno and A. McCallum. Topic Models Conditioned on Arbitrary Features with Dirichlet-multinomial Regression. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI '08)*, 2008.

[52] B. Minaei-bidgoli, A. Topchy, and W. F. Punch. A comparison of resampling methods for clustering ensembles. In *International Conference on Machine Learning: Models, Technologies and Applications*, pages 939–945, 2004.

[53] R. M. Neal. Probabilistic inference using markov chain monte carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 1993.

[54] R. M. Neal. Markov Chain Sampling Methods for Dirichlet Process Mixture Models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

[55] R. M. Neal. Defining priors for distributions using dirichlet diffusion trees. Technical Report 0104, Dept. of Statistics, University of Toronto, 2001.

[56] R. M. Neal. Density modeling and clustering using dirichlet diffusion trees. *Bayesian Statistics*, 7:619–629, 2003.

[57] X. Ning, H. Rangwala, and G. Karypis. Multi-assay-based structure activity relationship models: Improving structure activity relationship models by incorporating activity information from related targets. *Journal of Chemical Information and Modeling*, 49(11):2444–2456, 2009. PMID: 19842624.

[58] Y. Okuno, J. Yang, K. Taneishi, H. Yabuuchi, and G. Tsujimoto. GLIDA: GPCR-ligand database for chemical genomic drug discovery. *Nucleic Acids Research*, 34(suppl1):D673–D677, 2006.

[59] O. Papaspiliopoulos and G. O. Roberts. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika*, 95(1):169–186, March 2008.

[60] J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900, 1997.

[61] K. Punera and J. Ghosh. Soft cluster ensembles. In J. V. de Oliveira and W. Pedrycz, editors, *Advances in Fuzzy Clustering and its Applications*, chapter 4, pages 69–90. John Wiley & Sons, Ltd, 2007.

[62] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: a supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[63] C. E. Rasmussen. The infinite gaussian mixture model. In *Advances in Neural Information Processing Systems (NIPS)*, volume 12, pages 554–560, 2000.

[64] A. Rodriguez, D. B. Dunson, and A. E. Gelfand. The nested Dirichlet process. *Journal of the American Statistical Association*, 103(483):1131–1154, September 2008.

[65] D. M. Roy and Y. W. Teh. The Mondrian process. In *Advances in Neural Information Processing Systems (NIPS)*, volume 21, 2008.

[66] J. B. Schafer, J. Konstan, and J. Riedi. Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, EC '99, pages 158–166, New York, NY, USA, 1999. ACM.

[67] S. L. Schreiber. Chemical genetics resulting from a passion for synthetic organic chemistry. *Bioorg Med Chem*, 6(8):1127–1152, Aug 1998.

[68] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[69] M. Shafiei and E. Milios. Latent Dirichlet co-clustering. In *IEEE International Conference on Data Mining*, pages 542–551, 2006.

[70] H. Shan and A. Banerjee. Bayesian co-clustering. In *IEEE International Conference on Data Mining (ICDM)*, 2008.

[71] A. Strehl and J. Ghosh. Cluster ensembles—A knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3, 2003.

[72] P. Symeonidis, A. Nanopoulos, A. Papadopoulos, and Y. Manolopoulos. Nearest-Biclusters Collaborative Filtering. In *WEBKDD'06*, Philadelphia, Pennsylvania, USA, August 2006.

[73] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[74] Y. W. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for hdp. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, 2008.

[75] Y. W. Teh, D. Newman, and M. Welling. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Advances in Neural Information Processing Systems (NIPS)*, volume 19, 2007.

[76] N. Tolliday, P. A. Clemons, P. Ferraiolo, A. N. Koehler, T. A. Lewis, X. Li, S. L. Schreiber, D. S. Gerhard, and S. Eliasof. Small molecules, big players: the national cancer institute's initiative for chemical genetics. *Cancer Res*, 66(18):8935–8942, Sep 2006.

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

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

[79] A. Topchy, E. Topchy, A. K. Jain, and W. Punch. Combining multiple weak clusterings. In *International Conference on Data Mining*, pages 331–338, 2003.

[80] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, Dec. 2007.

[81] R. Waagepetersen and D. Sorensen. A tutorial on reversible jump mcmc with a view toward applications in QTL-mapping. *International Statistic Review*, 69(1):49 – 61, 2001.

[82] N. Wale and G. Karypis. AFGEN. Technical report, Department of Computer Science & Enigneering, University of Minnesota, 2007. www.cs.umn.edu/ karypis.

[83] C. Wang and D. M. Blei. Variational Inference for the Nested Chinese Restaurant Process. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

[84] H. Wang, H. Shan, and A. Banerjee. Bayesian clustering ensembles. In *SIAM International Conference on Data Mining*, pages 211–222, 2009.

[85] P. Wang, C. Domeniconi, and K. B. Laskey. Latent dirichlet bayesian co-clustering. In *ECML PKDD '09: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 522–537. Springer-Verlag, 2009.

[86] Z. Xu, V. Tresp, K. Yu, and H. peter Kriegel. Infinite hidden relational models. In *Proceedings of the 22nd International Conference on Uncertainity in Artificial Intelligence (UAI)*, 2006.

[87] K. Yu, S. Yu, and V. Tresp. Dirichlet enhanced latent semantic analysis. In *International Conference on Artificial Intelligence and Statistics*, pages 437–444. Society for Artificial Intelligence and Statistics, 2004.

Curriculum Vitae

Pu Wang was born in the People's Republic of China. He received his Bachelor of Engineering in Mechanics from Beihang University in 2004, and his Master of Science in Computer Science from Peking University in 2007. He joined the Volgenau School of Engineering at George Mason University for PhD study in Computer Science in 2007.