ALIREZA BAKHTIARI

Analysis of the Dirichlet Process Mixture Model with Application to Dialogue Act Classification

Mémoire présenté
à la Faculté des études supérieures de l'Université Laval
dans le cadre du programme de maîtrise en informatique
pour l'obtention du grade de Maître ès sciences (M.Sc.)

DÉPARTEMENT D'INFORMATIQUE ET DE GÉNIE LOGICIEL
FACULTÉ DES SCIENCES ET DE GÉNIE
UNIVERSITÉ LAVAL
QUÉBEC

2011

Résumé

La reconnaissance des intentions de l'utilisateur est l'un des problèmes les plus difficiles dans la conception des systèmes de dialogues. Ces intentions sont généralement codés en termes d'actes de dialogue, où un rôle fonctionnel est attribué à chaque énoncé d'une conversation. L'annotation manuelle des actes de dialogue est généralement coûteuse et prends du temps, il y a donc un grand intérêt à plutôt annoter automatiquement des corpus de dialogue.

Dans ce mémoire, nous proposons une approche non paramétrique bayésienne pour la classification automatique des actes de dialogue. Nous utilisons les mélanges par processus de Dirichlet (DPMM), dans lesquels chacune des composantes est déterminée par une distribution de Dirichlet-multinomial. Deux nouvelles approches pour l'estimation des hyperparamètres dans ces distributions sont introduites. Les résultats de l'application de ce modèle au corpus DIHANA montre que la DPMM peut récupérer le nombre réel d'étiquettes en haute précision.

Abstract

Recognition of user intentions is one of the most challenging problems in the design of dialogue systems. These intentions are usually coded in terms of Dialogue Acts (Following Austin's work on speech act theory), where a functional role is assigned to each utterance of a conversation. Manual annotation of dialogue acts is both time consuming and expensive, therefore there is a huge interest in systems which are able to automatically annotate dialogue corpora.

In this thesis, we propose a nonparametric Bayesian approach for the automatic classification of dialogue acts. We make use of the Dirichlet Process Mixture Model (DPMM), within which each of the components is governed by a Dirichlet-Multinomial distribution. Two novel approaches for hyperparameter estimation in these distributions are also introduced. Results of the application of this model to the DIHANA corpus shows that the DPMM can successfully recover the true number of DA labels with high precision.

Acknowledgements

First and foremost, I need to thank my advisor, Brahim Chaib-draa, for all the time and effort he has invested in me and my project, and his guidance and support throughout these past few years. I know I am a much better scholar today than I was when I started my graduate studies, and I owe most of this growth to him. Many thanks to my lab mates, specially Hamid and Abdeslam for their inestimable academic advice and personal support from the very first day I began this journey. I am also grateful to Mehdi for his invaluable comments and suggestions during the whole time I was working on my thesis.

I am truly indebted to my beloved friends Amin, Nafis and Samim who have always been there for me, supported me, and been like a family to me. Last but most important, I wish to thank my brother Reza, my sisters Shiva and Maryam, and most of all my parents, Naier and Sadegh, for they have bore me, raised me, taught me, trusted in me, and loved me. To them I dedicate this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Language is the hallmark of the human species. Several animals are capable of producing sounds to communicate with other members of their community; however, there is a vast difference between the chirps of a sparrow or the howls of a wolf, and a human reciting verses of Hafez in front of a large audience. Natural selection has endowed us with the exquisite gift of speech, where we can use words and structures to convey complex meanings in social communications. By language we inform others of our intentions, partake in conversations, discuss the most intricate philosophical questions, and even scrutinize the nature of language itself.

So the study of language may give us a clue as to the understanding of human intelligence, and thus help us advance our knowledge on how to establish intelligence into non-human agents. Despite the staggering growth in many other areas of science, there seems to be not so much consensus as to why this peculiar human trait has evolved throughout history. Some like Dawkins [1982] believe that language evolved for manipulation and deception of other people. Others such as Bickerton [1992] and Chomsky [2002] think of language more as an adaptation which allowed reflection and self-talking rather than communication with others. Some even propose that language evolved as a direct substitute for grooming [Dunbar, 1998] or as a courtship device to demonstrate the fitness of our brains [Miller, 2000]. Be that as it may, it still seems perplexing that such a great faculty with all its variations has evolved when simpler and more efficient coding could have served the very same purpose.

Perhaps the most compelling explanation for the emergence of language is the most intuitive one, that the human language faculty has evolved for communication in a knowledge-using, socially interdependent lifestyle [Pinker et al., 1992; Pinker, 2003]. The enormous benefits that lingual communication has brought about leaves no doubt

that after its evolution, this trait would not perish away. Language made us capable of explaining what we had in our minds and to expect others to cooperate with us in order to achieve higher altruistic goals.

Human communication by means of language is done through *conversation* or *dialogue*. Whether it is for participating in business meetings, ordering food from a restaurant, or asking for fares and timetables at a train station, we all indulge in some sort of dialogue with other people to achieve a certain goal. So any non-human agent attempting to display intelligence in its interaction with human users must be capable of maintaining a conversation in a coherent manner.

These non-human intelligent agents, also called *conversational agents*, live in the constrained world of language where the only possible observations are utterances, and the only actions available to them are to generate utterances. Even though the agents designed so far are way too limited in their functionality, still any attempt to design a conversational agent relies on an analysis of how humans interact among themselves, and for this we have to know what specific properties does this social activity posses.

The most apparent characteristic of any lingual conversation is that it necessarily involves multiple agents. It is impossible to converse by speaking when there is no other party listening to what is being said. This joint activity is built up of consecutive turns, where each participant says something when it is his or her turn to speak. In the case where there are only two people conversing, this means that first, speaker A starts saying something, then speaker B replies by saying something relevant to what A said, then the turn is passed to speaker A, and so on and so forth.

Even though it might be taken for granted, but communication (either by dialogue or by any other means) can only occur when both agents are able to recognize what the other person is saying. Consider the following example from Allen [1987] where Jack and Sue are trying to work on a communication scheme where Jack leaves the window of his room open when it is safe for Sue to visit him. This means that by seeing the open window, Sue can come visit him and they have been able to successfully communicate with one another. Now, assume that it is a hot summer day and Jack decides to open the window, not as a communication signal, but as to let fresh air into the room. Sue sees the open window and decides it is safe to visit. In this case, Jack and Sue have failed to communicate with one another because Sue has misinterpreted the signal and has not been able to recognize the true intention of Jack. On the other hand, suppose Jack opens the window and says to himself that although it is a hot summer day, he wants Sue to come by. Now if Sue interprets the open window as an act of cooling the room, she has been unable to understand the intention of Jack and once again, they

have failed to communicate with each other.

It thus follows that *communication can occur only when both agents are able to understand the true intention of one another.* In the domain of linguistic conversations, these intentions are captured using units called *speech acts*, or in more functional terms, *dialogue acts*. In the next section, we will briefly explain these linguistic units and explain their application in dialogue analysis.

1.1 Dialogue Acts

Suppose a traveller goes to a railway station information booth and asks '*I want to know the fare for the train leaving at 7:45 going to Barcelona*'. This simple utterance can be easily understood by a human agent and he can provide the costumer with appropriate information regarding train fares. However, if we are designing a conversational agent to do this task, our agent needs to know many things in advance before being able to respond to the costumer. It must be able to recognize the names, numbers, or any other relevant information in the utterance, perhaps know the context to some extent, and generate an utterance in response to the user. But first and foremost, in order to make an apt decision it needs to realize the intention of the user by understanding what the utterance is really about. These higher level information units are called *Dialogue Acts*.

In more precise terms, dialogue acts are linguistic units that represent the intention hidden in each utterance of a dialogue by assigning a functional role to that utterance. For example, the utterance above may be given a simple DA tag such as <Question>, or a more insightful label such as <Question-Fare-Departure Hour>, based on what annotation scheme has been used. However, recognition of these units is one of the main challenges in the development of conversational agents.

To begin with, a single utterance may be used to formulate various intentions. For example, a simple utterance such as 'Okay' could serve as an acknowledgement, an attention grabber, an acceptance, a yes answer or even as a conversation starter. Second of all, as we mentioned earlier, different annotation schemes yield different annotation tags. Some schemes like the SWBD-DAMSL are very general, such that they can be easily used for various contexts, whereas others like DIHANA are precise but task limited.

But perhaps the most intricate of all revolves around how our conversational agent should recognize these tags. It doesn't seem feasible to inject these annotations manu-

ally because utterances will definitely vary in real life situations and it is impossible to annotate all utterances beforehand. So we must resort to some automatic dialogue act classification techniques to overcome this difficulty.

To date, several methods have been applied to the automatic recognition of dialogue acts, among which include the use of sequence n-gram models [Nagata & Morimoto, 1994; Reithinger & Maier, 1995], application of hidden Markov models [Stolcke et al., 2000], the naive Bayes classifier [Grau et al., 2004] and the multilayer perceptron [Sanchis & Castro, 2002; Levin et al., 2003]. While some of these approaches, or combinations of them, have proven to yield successful results, but they seem to be limited in the sense that they are all parametric; i.e. they all require the number of Dialogue acts to be determined a priori. This information is almost never available, because conversational agents never know what situations they might encounter during a dialogue. This limitation asks for more flexible methods that can learn model parameters by themselves.

1.2 Nonparametric Bayesian Methods

Parametric models are bound to a finite number of parameters that are not allowed to grow or shrink over time. Examples of these include document clustering or dialogue act classification when the number of components are fixed by the model. To surpass this functional limitation, *nonparametric Bayesian methods* allow models that are not limited to finite parametrizations and 'let the data speak' themselves. This means that, for example, while we may start off with a specific number of DA labels, these components may increase or decrease in number during the course of the execution.

To make this possible, our models are no longer limited to parametric distributions; they are based on *stochastic processes*. In simple terms, a stochastic process is an indexed collection of random variables where the index set is allowed to be infinite [Blei et al., 2010]. Among these structures, *Dirichlet processes*, and infinite mixture models based on it, are one of the most important nonparametric structures used in machine learning applications. We can think of the Dirichlet process as the infinite dimensional counterpart of the general Dirichlet distribution. Using these models, our parameters can now range over infinite dimensional spaces and allow for structures with open ended cardinality.

1.3 Main Contributions

The primary contribution of this thesis is to offer a nonparametric Bayesian approach to the problem of automatic dialogue act recognition in DIHANA; a corpus composed of 900 dialogues about railway information such as trip timetables and prices. As mentioned earlier, previous attempts to the problem of automatic classification were all grounded on parametric models, which meant that the number of dialogue acts had to be determined when constructing the model. By using nonparametric Bayesian models, and the Dirichlet process in particular, we achieve more flexible models where the data are let to speak for themselves. In other words, our model finds the number of DA's necessary for capturing all information in the corpus by itself, and assigns each utterance of the corpus to one of these DA labels.

The model proposed here is an infinite mixture model, called the Dirichlet Process mixture model, in which the number of components (dialogue acts) grow as more data are observed. Each of these (potentially) infinite number of components is governed by a Dirichlet-Multinomial distribution, and utterances are assigned to components based on the word counts in each of them. The second main contribution of this thesis, which is fully explained in Chapter 4, is to provide two novel approaches to hyperparameter estimation in the Dirichlet-Multinomial distribution by addressing an important implementation shortcoming in the previous estimation techniques. The methods are later used in our experiments for updating the parameters of each DA component.

1.4 Thesis Outline

This thesis is organized as follows. In Chapter 2, we provide a background on some of the most important mathematical concepts that will be used in the later chapters, such as the basics of measure theory, Bayesian statistics and exponential families, and the most prominent methods for approximate inference in the Bayesian framework. Chapter 3 then proceeds with an in depth analysis of nonparametric Bayesian methods and in particular, the Dirichlet process. Among the various representations of this process, attention is focused on the Chinese Restaurant Process and the important properties that stem from this representational metaphor. We present our novel hyperparameter estimation techniques for the Dirichlet-Multinomial distribution in Chapter 4, along with other estimation methods for infinite dimensional mixture models. In Chapter 5, we give an overview of previous attempts to automatic dialogue act recognition, and propose the nonparametric model along with results of applying the method on the

DIHANA corpus. Finally, Chapter 6 concludes with a summary of our contributions and a consideration for future research.

Chapter 2

Background

In this background chapter, we review some of basic mathematical and statistical methodologies upon which our work is based. We begin by giving a basic introduction to measure theory and σ-algebras, and then describe basics of Bayesian inference and families of probability distributions, respectively. The last section of this chapter will be devoted to approximate methods for Bayesian inference which provide alternatives to exact inference in cases where it is impossible to provide analytic solutions to this type of inference.

2.1 Basics of Measure Theory

Measure theory [Billingsley, 1995; Athreya & Lahiri, 2006] is the study of measures, which intuitively aims at generalizing the notion of length to any suitable subset. In other words, what we are looking for is a concept of length that can include as many subsets of $\mathbb{R}$ as possible. This length $\mu(A)$, that we assign to a bounded subset A of $\mathbb{R}$, should have the following properties:

I. Length: The value $\mu(A)$ assigned to an open or closed interval is its length, i.e. if $A = (a, b)$ or $A = [a, b]$, then $\mu(A) = b - a$.

II. Translation invariance: If $c \in \mathbb{R}$, then $\mu(A + c) = \mu(A)$ where $\mu(A + c)$ denotes the set $\{x + c | x \in A\}$, i.e. if $A = (a, b)$, then $A + c = (a + c, b + c)$ and by the first property we have $\mu(A + c) = (b + c) - (a + c) = b - a = \mu(A)$.

III. Countable additivity: For any countable collection of bounded subsets $\{A_n\}_{n=1}^{\infty}$

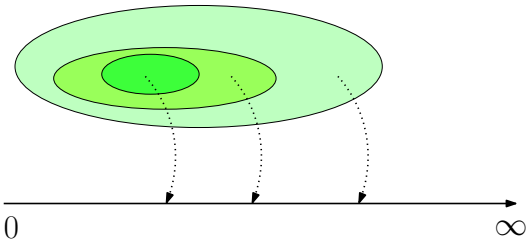

Figure 2.1: A measure is a function that assigns real numbers to subsets of a σ-algebra

of $\mathbb{R}$, we have:

$$\mu(\bigcup_{n=1}^{\infty} A_n) \leq \sum_{n=1}^{\infty} \mu(A_n)$$

and we have equality when the sets are pairwise disjoint. For instance, suppose $n = 2$ and the sets are defined as $A_1 = (a, b)$ and $A_2 = (b - \epsilon, c)$ where $\epsilon \geq 0$. Then the right hand side of the inequality $c - a + \epsilon$ would be greater than the left side which is $c - a$.

IV. Monotonicity: If $A \subset B$, then $\mu(A) \leq \mu(B)$, i.e. if $A = (a, b)$ and $B = (a, c)$ where $c \geq b$, then it is clear that $A \in B$ and we have $\mu(A) = b - a \leq \mu(B) = c - a$.

Unfortunately, it seems that it is not possible to have a μ defined over all subsets of real numbers satisfying properties I-IV (see e.g. Franks [2009] or Loève [1977]). Instead with a little modification, we can define a function that covers not all subsets, but a set of subsets in $\mathbb{R}$. This collection is called σ-algebra and is defined as follows:

Definition 2.1. A subset Σ of the power set of a set Θ is a σ-algebra if it has the following properties:

I. Σ is not empty;

II. Σ is closed under complements; in other words If E is in Σ, then its $(X - E)$ is also in Σ;

III. Σ is closed under countable unions: $A_1, A_2, \ldots, \in \Sigma$ then $\cup A_i \in \Sigma$.

All elements $A \in \Sigma$ of the σ-algebra are called *measurable sets*, and the ordered pair (Θ, Σ) where Θ is a set and Σ is a σ-algebra over Θ, is called a *measurable space*. To make this clearer, consider the set $\Theta = \{a, b, c, d\}$. One possible sigma algebra on Θ would be $\Sigma = \{\emptyset, \{a, b\}, \{c, d\}, \{a, b, c, d\}\}$ which satisfies the conditions I-III of Definition 2.1.

The function that assigns real numbers to subsets of a σ-algebra is called a *measure*. In technical terms, a measure is defined as follows:

Definition 2.2. A function $\mu : \Sigma \to [0, \infty]$ over the measure space (Θ, Σ) is called a *measure* if it has the following three properties:

I. $\mu(A) \geq 0$ for all $A \in \Sigma$,

II. $\mu(\emptyset) = 0$,

III. $\mu(\cup(A_i)) = \sum(\mu(A_i))$.

Various measures have been defined for different contexts. A *counting measure* [Banuelos, 2003], for example, is defined by $\mu(A) =$ number of elements in A. Perhaps the most important measure function is the *probability measure*. A probability measure is simply a measure with total measure equal to one (i.e. $\mu(\Theta) = 1$). In a similar way, a *probability space* is a measure space over probability measures and is denoted by the triple (Θ, Σ, P) where Θ is the measure space, Σ is a σ algebra, and P is a probability measure.

2.1.1 Probability Spaces and Random Variables

A probability space [Loève, 1977; Billingsley, 1995] is a mathematical construct that models an experiment consisting of states that occur randomly. It is denoted by a triple (Θ, Σ, P) and consists of the following parts:

I. Sample space Θ which is the set of all possible outcomes of an experiment.

II. σ-algebra Σ which is a collection of events. Basically, it is the set of outcomes that we are interested in and must be a subset of Θ.

III. Probability measure P which assigns a real number to the events. Since a probability measure has total measure equal to one, then according to property III in Definition 2.2, any probability measure must be a real number between zero and one. This number is referred to as a *probability assignment*, or simply a *probability*.

A *random variable* is simply a function that assigns a number to every outcome of an experiment. The following definition gives a formal account of random variables in real observation spaces.

Definition 2.3. The function $X : \Theta \to \mathbb{R}$ is a real-valued random variable over the probability space (Θ, Σ, P) if:

$$\{\omega : X(\omega) \leq r\} \in \Sigma \qquad \forall r \in \mathbb{R} \tag{2.1}$$

Random variables can be any numerical quantity of interest in an experiment. As an example, when tossing a perfect coin we can describe the number of heads appearing as a random variable X given as follows:

$$X = \begin{cases} 1 & \text{if heads,} \\ 0 & \text{if tails.} \end{cases}$$

2.1.2 Probability Distribution and Density Functions

Let (Θ, Σ, P) be a probability space and X be a random variable over the sample space Θ. Consider a σ-algebra Σ defined by $\{X \leq x\}$. This means that we are only interested in the outcomes that are less than or equal to a specific number x.

The elements of Θ that are contained in the event $\{X \leq x\}$ change as the number x changes. Therefore, the number (probability) that the probability measure $P(\{X \leq x\})$ assigns to the event $\{X \leq x\}$ depends on the value of x. This number is denoted by $F(x)$ and is called the *cumulative distribution function* or simply the *distribution function* of the random variable X.

The derivative of a distribution function $dF(x)/dx$ is denoted by $f(x)$ and called the *probability density function*. This function describes the relative likelihood for a continuous random variable to occur at a given point in the space. For instance, the *uniform distribution* on $(0, 1)$ is defined as:

$$F(x) = \begin{cases} 0 & x \leq 0 \\ x & 0 \leq x \leq 1 \\ 1 & x > 1 \end{cases}$$

The probability density function of the uniform distribution will then be $f(x) = 1$ for $x \in (0, 1)$ and 0 otherwise. This means that all intervals with the same length on $(0, 1)$ will be equally probable.

In the case of discrete random variables, the density function $f(x)$ will have the form of a collection of positive discrete masses. In such cases, $f(x)$ represents the probability that the random variable is exactly equal to some specific value and is usually called the *probability mass function*.

2.1.3 Expectation and Variance

Expected value is one of the fundamental concepts in probability. It is formally defined as follows:

Definition 2.4. If X is a random variable defined over a probability space (Θ, Σ, P), then the expected value of X, denoted by $E(X)$ is defined as:

$$E(X) = \int_{\Theta} X dP \tag{2.2}$$

Therefore, for real-valued random variables the expected value becomes:

$$E(X) = \int_{-\infty}^{+\infty} x f(x) dx \tag{2.3}$$

The *variance* of a random variable, denoted by $Var(X)$, is the deviation of that variable from its expected value. In other words:

$$Var(X) = E((X - E(X))^2) = E(X^2) - (E(X))^2 \tag{2.4}$$

The basics that we introduced in this section lay the mathematical foundation of the rest of the topics that we will discuss. In the next section, we will overview one of the prominent interpretations of probability measures which assumes that our subjective uncertainties can be expressed in terms of probabilities. This simple expression forms the backbone of *Bayesian statistics*.

2.2 Bayesian Statistics

The Bayesian interpretation of probability can be seen as an extension of logic to describe *degrees of belief* [Wallach, 2008]. Any set of beliefs can be mapped onto probabilities as long as they satisfy Cox's axioms of consistency (see Cox [1946]). Bayesian

probability enables us to quantify all kinds of uncertainty in our statistical inferences when we are concerned with drawing conclusions, from observed data, about quantities that are not observed. This approach is, for instance, extremely useful in modeling spoken and text data where the true nature of the underlying topics, syntactic structure and dialogue acts are unknown.

Before proceeding, we should clarify some of the basic notions that we will be dealing with in any statistical evaluation. As normal with many disciplines of science, we will describe the system that governs a random variable using a *mathematical model*, i.e. a set of equations describing the underlying distributions from which we assume our dataset is sampled. For example, tossing a standard coin is normally described by a *binomial distribution*[1]. In statistical modelling, we should distinguish between two types of unobserved data that we are trying to draw conclusions. First, potentially observable quantities, such as the next flipping of a coin; and second, unobservable quantities that govern the underlying model which leads to the observed data. In the same example, the probability of observing 'heads' is the unobserved quantity or *parameter* when the underlying model has a binomial distribution. The parameters of the model are generally denoted by the vector $\boldsymbol{\theta}$[2].

2.2.1 Bayes' theorem

Bayesian statistical inferences about the unobserved data or the parameter vector $\boldsymbol{\theta}$ are made using *Bayes' theorem* [Gelman et al., 2004; Bolstad, 2010]. This rule which underlies most modern AI systems for probabilistic inference, is explained in the following theorem:

Theorem 2.1. *Let (Θ, Σ, P) be a probability space and let $A, B \in \Sigma$ be such that $P(A) > 0$ and $P(B) > 0$. Then:*

$$P(A|B)P(B) = P(B|A)P(A) \tag{2.5}$$

where $P(A|B)$ is the conditional probability of the event A given event B and defined as:

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(A \cap B)}{P(B)} \tag{2.6}$$

[1]The binomial distribution is a special case of the multinomial distribution where there are only two possible outcomes. See Section 2.3.2 for more details.

[2]Throughout this thesis bold lower-case letters are meant to be vectors, bold upper-case letters are matrices, and italic letters refer to scalar values.

Suppose we have n observations from a statistical experiment (for example n trials of a coin flip) and denote them by the vector $\mathbf{y} = (y_1, \ldots, y_n)$. For the purpose of statistical inference, these observations are often assumed to be *independent and identically distributed* (i.i.d)[Gelman et al., 2004]; their joint probability $p(y_1, \ldots, y_n)$ is equal to $\prod_{i=1}^{n} p(y_i)$ and the underlying distribution function which has generated them is the same[1]. Supposing that the model is governed by the parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)$, we can generalize Theorem 2.1 and write the theorem in its unscaled form as:

$$p(\boldsymbol{\theta}|\mathbf{y}) \propto p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta}) \tag{2.7}$$

The reason for writing Bayes' theorem in proportional form (i.e. using $\propto$) is that in most cases, as we will further explain in Section 2.4, the exact computation of the posterior is computationally infeasible due to the intractability of the scaling factor for large dimensions. Therefore, many approximate methods rely just on its shape, as described in Eq. (2.7), to get as close as possible to the true value of the posterior distribution.

Under the Bayesian approach, we assume that while we do not know the true value of the parameter vector $\boldsymbol{\theta}$ we can place probabilities on the various possible $\boldsymbol{\theta} \in \Sigma$. The density function $p(\boldsymbol{\theta})$ in Eq. (2.7) returns this *prior belief* in the parameter vector.

The observed data $y_1, \ldots, y_n$ also convey information about the true value of the parameter through the *likelihood function* $p(y_1, \ldots, y_n|\theta_1, \ldots, \theta_p)$. This function gives relative weights to every possible parameter value that comes from the observations. Bayes' theorem allows us to combine these two sources of information (our prior and the likelihood of data) into the *posterior* density, which formulates how to consistently update our belief in the parameter vector after observing the data.

It must be noted that Eq. (2.7) does not give the exact posterior $p(\boldsymbol{\theta}|\mathbf{y})$, but it does give its shape. In order to find the actual posterior density, we must divide it by a scaling factor so that it integrates to 1. According to Bayes' rule, the scaling factor $p(\mathbf{y})$, which is called the *marginal distribution* [Gelman et al., 2004] of $\mathbf{y}$, is equal to:

$$p(\mathbf{y}) = \int_{\Theta} p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})d\boldsymbol{\theta} \tag{2.8}$$

In this case, Bayes' theorem as described in Eq. (2.7) can be written in its non propor-

[1] In practical applications, this condition may not be easily met. A sufficient and more realistic assumption would be to consider them as *exchangeable*. De Finetti [1931] showed that exchangeable random variables share the main properties of i.i.d variables. See Aldous [1985] or Fox [2009] for more explanation.

tional form as follows:

$$p(\boldsymbol{\theta}|\mathbf{y}) = \frac{p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})}{p(\mathbf{y})} = \frac{p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})}{\int_\Theta p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})d\boldsymbol{\theta}} \tag{2.9}$$

2.2.2 Prediction

One of the main goals in Bayesian statistical analysis is to make predictions about an unknown observable based on the data that we have already observed. Suppose we have made n observations $\mathbf{y} = (y_1, \ldots, y_n)$ and we are interested in the probability distribution of a future observation y_{n+1}. This *predictive distribution* ([Aitchison & Dunsmore, 1975; Bolstad, 2010; Gelman et al., 2004]) can be found by integrating the parameter vector out of the joint posterior distribution:

$$p(y_{n+1}|y_1, \ldots, y_n) = \int_\Theta p(y_{n+1}, \boldsymbol{\theta}|y_1, \ldots, y_n)d\boldsymbol{\theta} \tag{2.10}$$

We can further simplify Eq. (2.10) by applying the following lemma which can be simply obtained from the definition of conditional probability in Eq. (2.6):

Lemma 2.1. *For any set of events A, B and C, we have:*

$$p(A, B|C) = p(A|B, C)p(B|C) \tag{2.11}$$

Now by applying Lemma 2.1 to Eq. (2.10) and then applying Bayes' theorem in its complete scaled form, i.e. $p(\boldsymbol{\theta}|\mathbf{y}) = \frac{p(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})}{p(\mathbf{y})}$, it follows that:

$$p(y_{n+1}|y_1, \ldots, y_n) = \int_\Theta p(y_{n+1}|y_1, \ldots, y_n, \boldsymbol{\theta})p(\boldsymbol{\theta}|y_1, \ldots, y_n)d\boldsymbol{\theta} \tag{2.12}$$

$$= \frac{\int_\Theta p(y_{n+1}|\boldsymbol{\theta})p(y_1, \ldots, y_n|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}{p(\mathbf{y})} \tag{2.13}$$

where $p(\mathbf{y})$ is the marginal distribution of $\mathbf{y}$ as defined in Eq. (2.8). The last equation follows because in our model, y_{n+1} and $\mathbf{y}$ are conditionally independent given the parameter vector $\boldsymbol{\theta}$, in other words:

$$p(y_{n+1}|y_1, \ldots, y_n, \boldsymbol{\theta}) = p(y_{n+1}|\boldsymbol{\theta}) \tag{2.14}$$

.

2.2.3 Parameter Estimation

There are several methods available to estimate the parameters of a statistical model governing a set of observable outcomes. Here we will overview some of the most important approaches which are used in the next chapters.

In *Maximum Likelihood Estimation* (MLE) [Alpaydin, 2004; Robert & Casella, 2004], we are interested in finding the parameter value $\hat{\boldsymbol{\theta}}_{MLE}$ that makes our observations the most likely to be drawn. Normally, the calculation is simplified by maximizing the logarithm of the likelihood, the *log likelihood* function:

$$\hat{\boldsymbol{\theta}}_{MLE} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \{\log p(\mathbf{y}|\boldsymbol{\theta})\} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \{\log p(y_1, \ldots, y_n|\theta_1, \ldots, \theta_p)\} \tag{2.15}$$

Because the logarithm function is monotonically increasing, the logarithm of a function has the same maxima as the function itself. There are two main motivations for using the log likelihood function; First, under the i.i.d assumption, the right hand side of Eq. (2.15) becomes $\sum_{i=1}^{n} \log p(y_i|\boldsymbol{\theta})$ which is more convenient to deal with, since the derivative of a sum of terms is easier to compute than the derivative of a product (this comes in handy when we are trying to find the extrema of a function). Second, many of the common probability distributions belong to the exponential family, and the logarithm of such distributions are much easier to differentiate than the original function.

The main problem with MLE is that it generally has a poor predictive performance since it highly overfits the available observations [Bishop, 2006]. For instance, suppose we flip a coin three times and observe heads each time. A classical maximum likelihood estimate of the probability of landing heads would give 1, which means that all future tosses will land heads.

Now assume that we also have a prior distribution over our parameter vector $\boldsymbol{\theta}$. The method of *Maximum A posteriori Estimation* (MAP) [Alpaydin, 2004] then tries to find the parameter vector $\hat{\boldsymbol{\theta}}_{MAP}$ that maximizes the posterior density as defined in Eq. (2.7). Again, using the logarithm of the posterior is a more convenient choice:

$$\begin{aligned} \hat{\boldsymbol{\theta}}_{MAP} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \{\log p(\boldsymbol{\theta}|\mathbf{y})\} &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \{\log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) - \log p(\mathbf{y})\} \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \{\log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})\} \end{aligned} \tag{2.16}$$

where the last equation follows from the fact that $p(\mathbf{y})$ is constant for all values of $\boldsymbol{\theta}$, and thus does not affect the argmax function. When we have no prior information

about the parameters, then the posterior will have the same form as the likelihood and our MAP estimate will be equal to the MLE estimate of $\boldsymbol{\theta}$.

The main issue with MLE and MAP is that they are *point estimators*; they use the whole dataset to calculate the value of a single variable, and for this reason they are not good representatives for the Bayesian framework. However, in *Bayesian estimation*, the data are summarized by the use of distributions instead of single points, and thus they try to estimate the mean value of the distribution along with *credible intervals*.

Whereas most of the difficulties related to MLE and MAP are optimization problems, like solving for the maxima or dealing with local extrema, Bayesian estimation normally results in integration problems [Robert & Casella, 2004]. Bayes' estimator [Alpaydin, 2004] is defined as the expected value of the posterior density:

$$\hat{\boldsymbol{\theta}}_{Bayes} = E(\boldsymbol{\theta}|\mathbf{y}) = \int_{\Theta} \boldsymbol{\theta} p(\boldsymbol{\theta}|\mathbf{y}) d\boldsymbol{\theta} \tag{2.17}$$

The motivation behind taking the expected value is that the best estimate of a random variable is its mean. In other words, the mean square error is minimal if we chose the mean vector as our estimate for $\boldsymbol{\theta}$.

For means of mathematical simplification, a conjugate prior (as explained in Section 2.3.1) is usually chosen for the prior distribution. However, in some situations the choice of a conjugate prior may not be realistic and can imply restrictions on the modeling of the prior information. In such cases numerical sampling methods based on *Monte Carlo integration techniques*, which will be further explained in the Section 2.4, should be used to evaluate the integral of Eq. 2.17.

2.3 Exponential Families

Exponential families [Hazewinkel, 2002; Clark & Thayer, 2004; Bishop, 2006] are a broad class of probability distributions which have many important properties in common. Many of the most widely used distributions, including the normal, gamma, Dirichlet, multinomial and Poisson belong to this class of distributions.

Definition 2.5. A parametrized class of distributions $\mathbb{F}$ is a k parameter exponential family if each member of the family has a density of the form:

$$p(\mathbf{y}|\boldsymbol{\theta}) = h(\mathbf{y}) g(\boldsymbol{\theta}) \exp(\boldsymbol{\eta}(\boldsymbol{\theta}) \cdot \mathbf{t}(\mathbf{y})) \tag{2.18}$$

where $\boldsymbol{\eta}(\boldsymbol{\theta}) = (\eta_1(\boldsymbol{\theta}), \eta_2(\boldsymbol{\theta}), \ldots, \eta_k(\boldsymbol{\theta}))$ is the *natural parameter* of the distribution, $\mathbf{t}(\mathbf{y}) = (t_1(\mathbf{y}), t_2(\mathbf{y}), \ldots, t_k(\mathbf{y}))$ the *sufficient statistic*, and $h(\mathbf{y})$ and $g(\boldsymbol{\theta})$ are some functions of $\mathbf{y}$ and $\boldsymbol{\theta}$, respectively[1].

Note that the function $g(\boldsymbol{\theta})$ is automatically determined once the other functions have been chosen. We can interpret $g(\boldsymbol{\theta})$ as the normalization factor that ensures the probability density integrates to 1:

$$g(\boldsymbol{\theta}) = \left(\int h(\mathbf{y}) \exp(\boldsymbol{\eta}(\boldsymbol{\theta}) \cdot \mathbf{t}(\mathbf{y})) d\mathbf{y} \right)^{-1} \tag{2.19}$$

A *sufficient statistic* [Fisher, 1922; Hazewinkel, 2002] is a measure such that "no other statistic[2] which can be calculated from the same sample provides any additional information as to the value of the parameter"[Fisher, 1922]. In other words:

$$p(\mathbf{y}|\mathbf{t}(\mathbf{y}), \boldsymbol{\theta}) = p(\mathbf{y}|\mathbf{t}(\mathbf{y})) \tag{2.20}$$

To see why $\mathbf{t}(\mathbf{y})$ is the sufficient statistic for the exponential family, consider a set of i.i.d. observations $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$, according to which we are trying to find the MLE of our parameter vector $\boldsymbol{\theta}$. The likelihood function with respect to the parameter $\boldsymbol{\theta}$ is given by:

$$p(\mathbf{Y}|\boldsymbol{\theta}) = \left(\prod_{i=1}^{n} h(\mathbf{y}_i) \right) g(\boldsymbol{\theta})^n \exp \left\{ \boldsymbol{\eta}(\boldsymbol{\theta}) \cdot \sum_{i=1}^{n} \mathbf{t}(\mathbf{y}_i) \right\} \tag{2.21}$$

The maximum value for $\boldsymbol{\theta}$ is obtained by setting the gradient of $\log p(\mathbf{Y}|\boldsymbol{\theta})$ in Eq. (2.21) to zero, with respect to $\boldsymbol{\theta}$:

$$\frac{\nabla g(\hat{\boldsymbol{\theta}})}{g(\hat{\boldsymbol{\theta}})} + \nabla \boldsymbol{\eta}(\hat{\boldsymbol{\theta}}) \cdot \frac{\sum_{i=1}^{n} \mathbf{t}(\mathbf{y}_i)}{n} = 0 \tag{2.22}$$

Therefore the solution for the maximum likelihood estimator depends not on the individual observations but only on $\sum_{i=1}^{n} \mathbf{t}(\mathbf{y}_i)$. In other words, all of the useful information

[1] In this section, we slightly deviate from the notations that we have been using so far. In order to define the exponential family in its most general form, we have let each single observation to be a vector of values. This means that $\mathbf{y}$ represents a single observation vector, and $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ will represent the set of observations.

[2] A statistic is a function of the sample data which assigns a single measure to the whole data set. An example could be the arithmetic mean of the sample.

that our dataset can provide to help us in estimating the parameter vector is captured in the sufficient statistic of the distribution.

One of the main reasons why distributions from the exponential family are widely common in statistics comes from a theorem proven independently by Pitman [1936], Koopman [1936] and Darmois [1935] almost all at the same time. Essentially, what they have shown is that except for a few families such as uniform distributions, it is only in the distributions belonging to this family that the dimension of the sufficient statistic is bounded even as the number of samples grows to infinity (see Jeffreys [1961] or Barankin & Maitra [1963] for a more detailed explanation of the theorem). This is an important property for the exponential family because it leads to efficient parameter estimation methods.

2.3.1 Conjugate Distributions

In a Bayesian context, if the posterior distribution $P(\boldsymbol{\theta}|\mathbf{y})$ has the same functional form as the prior probability distribution $P(\boldsymbol{\theta})$, then the prior is called a *conjugate prior* [Hazewinkel, 2002; Robert & Casella, 2004] for the likelihood $P(\mathbf{y}|\boldsymbol{\theta})$. For reasons related to the Pitman-Koopman-Darmois theorem, conjugate priors are only to be found among distributions belonging to the exponential family (see e.g. Brown [1986]). For such distributions, the conjugate prior takes the following form:

$$p(\boldsymbol{\theta}|\boldsymbol{\xi},\nu) = f(\boldsymbol{\xi},\nu)g(\boldsymbol{\theta})^{\nu}\exp(\boldsymbol{\eta}(\boldsymbol{\theta})\cdot\boldsymbol{\xi}) \tag{2.23}$$

where $\boldsymbol{\xi}$ and ν are hyper-parameters that govern our prior distribution, and $f(\boldsymbol{\xi},\nu)$ is a normalizing constant ensuring that our probability distribution integrates to 1. To see that the posterior and prior actually have the same functional form, we should multiply this prior by the likelihood function obtained in Eq. (2.21):

$$\begin{aligned} p(\boldsymbol{\theta}|\mathbf{Y},\boldsymbol{\xi},\nu) &= p(\mathbf{Y}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\xi},\nu) && (2.24)\\ &\propto g(\boldsymbol{\theta})^{n+\nu}\exp\left\{\boldsymbol{\eta}(\boldsymbol{\theta})\cdot\left(\boldsymbol{\xi}+\sum_{i=1}^{n}\mathbf{t}(\mathbf{y}_i)\right)\right\} && (2.25)\end{aligned}$$

Which confirms the conjugacy of the posterior with our prior. Essentially, conjugate priors are chosen because of their flexibility; if the likelihood does not belong to the exponential family, then no conjugate prior exists and we should resort to numerical techniques for estimating the posterior [Huang, 2005].

In the following two sections, we will overview two of the most important members of the exponential family which will be used extensively in the next chapters.

2.3.2 Multinomial Distribution

Suppose we run n independent trials of an experiment such that in each trial, there are K possible outcomes and the probability of observing each of these outcomes is constant. A typical example is rolling a fair die 10 times, in which case there are 6 possible outcomes each occurring equally probable, i.e. $p(1) = p(2) \ldots = p(6) = \frac{1}{6}$. Under such circumstances, the experiment is governed by a *multinomial distribution*.

To be more formal, consider a sequence of independent and identical trials $Y_1, Y_2, \ldots, Y_n$ over the probability space (Θ, Σ, π)[1] with sample space $\Theta = \{1, \ldots, K\}$. The probability that our string of observations taking values $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ is governed by a K-dimensional *multinomial distribution* [Kotz et al., 2000; Balakrishnan & Nevzorov, 2003] and is given by:

$$p(y_1, y_2, \ldots, y_n|\boldsymbol{\pi}) = \frac{n!}{\prod_{i=1}^{K} n_i} \prod_{i=1}^{K} \pi_i^{n_i} \qquad n_i = \sum_{j=1}^{n} \delta(y_j, i) \tag{2.26}$$

where $\delta(x, y)$ is the Kronecker delta function; $\delta(x, y) = 1$ if $x = y$, and zero otherwise. In the special case where the sample space has exactly two members, the distribution is also called a *binomial distribution*. It can be readily shown that the multinomial distribution is a member of the exponential family. To see this, consider the multinomial distribution for a single observation. For means of convenience, we will represent the observation by a K-dimensional vector $\mathbf{x} = (x_1, x_2, \ldots, x_K)$ where x_i equals 1 only if the outcome of the trial is i, and zero otherwise. Therefore:

$$p(\mathbf{x}|\boldsymbol{\pi}) = \prod_{i=1}^{K} \pi_i^{x_i} = \exp\left\{\sum_{i=1}^{K} x_i \log \pi_i\right\} \tag{2.27}$$

Now comparing with Eq. (2.18), by replacing $\boldsymbol{\theta}$ with $\boldsymbol{\pi}$, we have:

$$\begin{cases} h(\mathbf{x}) = 1 \\ g(\boldsymbol{\pi}) = 1 \\ \boldsymbol{\eta}(\boldsymbol{\pi}) = (\log \pi_1, \log \pi_2, \ldots, \log \pi_K) \\ \boldsymbol{t}(\mathbf{x}) = \mathbf{x} \end{cases} \tag{2.28}$$

Note that for multinomial observations, the conjugate prior must also belong to the exponential family and, according to Eq. (2.23) along with the correspondences in Eq. (2.28), it must be in the following form:

[1] In the following sections, we will denote the probability measure of the exponential family by the vector $\boldsymbol{\pi}$ in order to avoid confusion.

$$p(\boldsymbol{\pi}|\boldsymbol{\xi}) = f(\boldsymbol{\xi}) \exp(\boldsymbol{\eta}(\boldsymbol{\pi}) \cdot \boldsymbol{\xi}) = f(\boldsymbol{\xi}) \exp(\sum_{i=1}^{K} \xi_i (\log \pi_i)) \tag{2.29}$$

In the next section, we will see that the Dirichlet distribution has a form similar to Eq. (2.29). In multinomial distributions, the expected number of times that outcome i will be observed over n trials is $E(Y_i) = n\pi_i$, and its variance is $Var(Y_i) = n\pi_i(1 - \pi_i)$.

2.3.3 Dirichlet Distribution

The K-dimensional *Dirichlet distribution* [Kotz et al., 2000; Balakrishnan & Nevzorov, 2003; Fox, 2009] is the conjugate prior for the K-dimensional multinomial distribution. Let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$ be a point in the unit $K - 1$ simplex[1]. Any point in this unit simplex satisfies the following properties:

$$0 \leq \pi_i \qquad \textit{and} \qquad \sum_{i=1}^{K} \pi_i = 1 \tag{2.30}$$

The Dirichlet distribution for the point $\boldsymbol{\pi}$ is uniquely defined by a set of hyper-parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_K) > 0$ and can be written as follows:

$$p(\pi_1, \ldots, \pi_K|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_i(\alpha_i))}{\prod_i(\Gamma(\alpha_i))} \prod_{i=1}^{K} \pi_i^{\alpha_i - 1} \tag{2.31}$$

where $\Gamma(x)$ is the standard Gamma function[2]. When there are only two parameters, the distribution is also called a *Beta distribution*. By convention, the exponents of a Dirichlet distribution are defined to equal $\alpha_i - 1$ so that the expected value and variance have the following simple forms [Sudderth, 2006]:

$$E(\pi_i) = \frac{\alpha_i}{s} \qquad\qquad Var(\pi_i) = \frac{E(\pi_i)(1 - E(\pi_i))}{s + 1} \tag{2.32}$$

The parameter $s = \sum_i(\alpha_i)$ is typically referred to as the *concentration parameter* and controls how concentrated the distribution is around its expected value [Huang, 2005]. Fig. 2.2 plots a Dirichlet distribution over the 2-simplex for various settings of the concentration parameter.

[1] A simplex is a generalization of the notion of a triangle to arbitrary dimensions. The $K - 1$ simplex has K vertices.

[2] The Gamma function is an extension of the factorial function to complex numbers with real parts. It is defined via the integral $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt$. For positive integers, the integral amounts to $\Gamma(n) = (n - 1)!$

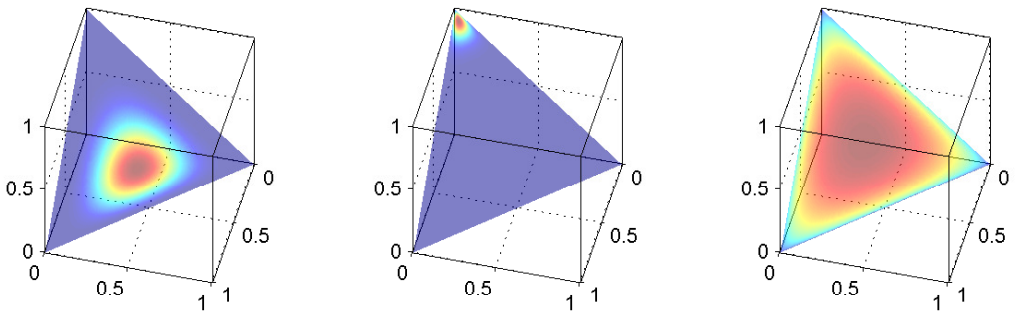

Figure 2.2: The Dirichlet distribution for different settings of the concentration parameter s. The plot on the right has a small s which results in a diffuse distribution, whereas the centre plot has a large s and therefore concentrated around the mean. Figure from Huang [2005]

The Dirichlet distribution is also a member of the exponential family. We can write Eq. (2.31) as:

$$p(\boldsymbol{\pi}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_i(\alpha_i))}{\prod_i(\Gamma(\alpha_i))} \exp\left\{\sum_{i=1}^{K}(\alpha_i - 1)\log \pi_i\right\} \tag{2.33}$$

Which corresponds with the standard formulation of members of the exponential family defined in Eq. (2.18) (with $\boldsymbol{\theta} = \boldsymbol{\alpha}$) as:

$$\begin{cases} h(\boldsymbol{\pi}) = 1 \\ g(\boldsymbol{\alpha}) = \dfrac{\Gamma(\sum_i(\alpha_i))}{\prod_i(\Gamma(\alpha_i))} \\ \boldsymbol{\eta}(\boldsymbol{\pi}) = (\log \pi_1, \log \pi_2, \ldots, \log \pi_K) \\ \boldsymbol{t}(\boldsymbol{\alpha}) = \boldsymbol{\alpha} - \mathbf{1} \end{cases} \tag{2.34}$$

Now, by setting $\boldsymbol{\xi} = \boldsymbol{\alpha} - \mathbf{1}$ and $f(\boldsymbol{\xi}) = g(\boldsymbol{\alpha})$ in Eq. (2.29), it can be readily seen that the Dirichlet distribution is indeed a conjugate prior for the multinomial distribution.

Consider a set of observations $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ from a multinomial distribution. The conjugacy of the Dirichlet distribution implies that if we assume a Dirichlet prior for $\boldsymbol{\pi}$; i.e. $\boldsymbol{\pi} \sim Dir(\alpha_1, \ldots, \alpha_K)$, then the posterior distribution is also a Dirichlet whichhas a

form similar to Eq. 2.25:

$$\begin{aligned} p(\boldsymbol{\pi}|y_1,\ldots,y_n,\boldsymbol{\alpha}) &\propto \left\{\boldsymbol{\eta}(\boldsymbol{\pi})\left((\boldsymbol{\alpha}-\mathbf{1})+\sum_{j=1}^{n}\mathbf{t}(y_j)\right)\right\} \\ &\propto \exp\left\{\sum_{i=1}^{K}(n_i+\alpha_i-1)\log\pi_i\right\} \\ &\propto Dir(\alpha_1+n_1,\ldots,\alpha_K+n_K) \end{aligned} \tag{2.35}$$

where $t_i(y_j)=\delta(y_j,i)$ and $n_i=\sum_{j=1}^{n}t_i(y_j)$. Therefore, we just need to update the vector of sufficient statistics with the number of observations of category k, as in Eq. (2.35).

When analysing the Dirichlet process, which we will cover in full detail in Chapter 3, it is useful to consider the aggregation and decimation properties of Dirichlet distributions. We will consider these properties below without proof. For a more thorough discussion and detailed proof, see Teh [2010] or Zhang [2008].

Theorem 2.2. *(Agglomeration) If* $(\pi_1,\pi_2,\ldots,\pi_K)\sim Dir(\alpha_1,\alpha_2,\ldots,\alpha_K)$*, then:*

$$(\pi_1+\pi_2,\pi_3\ldots,\pi_K)\sim Dir(\alpha_1+\alpha_2,\alpha_3,\ldots,\alpha_K) \tag{2.36}$$

and generally, if $(I_1,I_2,\ldots,I_s)$ *is a partition of* $\{1,2,\ldots,K\}$*, then:*

$$(\sum_{i\in I_1}\pi_i,\ldots,\sum_{i\in I_s}\pi_i)\sim Dir(\sum_{i\in I_1}\alpha_i,\ldots,\sum_{i\in I_s}\alpha_i) \tag{2.37}$$

Theorem 2.3. *(Decimation) If* $(\pi_1,\pi_2,\ldots,\pi_K)\sim Dir(\alpha_1,\alpha_2,\ldots,\alpha_K)$*, and* $(\tau_1,\tau_2)\sim Dir(\alpha_1\beta_1,\alpha_1\beta_2)$ *where* $\beta_1+\beta_2=1$*, then:*

$$(\pi_1\tau_1,\pi_1\tau_2,\pi_2,\ldots,\pi_K)\sim Dir(\alpha_1\beta_1,\alpha_1\beta_2,\alpha_2,\ldots,\alpha_K) \tag{2.38}$$

As we will see later, Dirichlet distributions play an important role in natural language modelling. The fact that they are the conjugate prior of the multinomial family makes them a computationally feasible choice for incorporating prior knowledge into our statistical models.

In many problems that arise in Bayesian statistics, one of the main challenges is to evaluate some sort of complex integral. Unfortunately, in most cases it is not possible to analytically calculate this value and we have to resort to other methods for overcoming this difficulty. The next section will introduce techniques that try to approximate the problem instead of directly calculating the exact value.

2.4 Approximate Inference

Two major classes of problems that arise in Bayesian inference are *integration* and *optimization* problems. In this section, we discuss approximate solutions to the first class; i.e. problems which involve evaluating the value of a complex integral in large dimensional spaces. In Bayesian statistics, we frequently encounter such integrals when dealing with posterior distributions. Examples include [Andrieu et al., 2003]:

- *marginalisation*; which involves calculating the marginal distribution of the observation vector by integrating the parameter vector out, as described in Eq. (2.8).
- *expectation*; from Eq. (2.2) we can find the expected value of a function $f(X)$ by:

$$E_P(f(X)) = \int_\Theta f(X)dP = \int_\Theta f(x)p(x)dx \tag{2.39}$$

  An example would be the computation of Bayes' estimator as defined in Eq. (2.17).
- *prediction*; as formulated in Eq. (2.12), which aims to predict the value of unknown observables based on what we have already observed.

Unfortunately, as we mentioned earlier, for many problems arising in practice exact solution of these integrals is intractable and so we have to rely on some sort of approximation. In general, approximation schemes fall into two broad categories, depending on whether they resort to deterministic or stochastic approximations [Bishop, 2006]. We will overview these two broad frameworks in the following sections.

2.4.1 Deterministic Approximation

Consider Bayes' theorem as described in Theorem 2.1. Normally, this theorem is expressed simply in the unscaled form *posterior* proportional to *prior* times *likelihood*, as formulated in Eq. (2.7). However, as mentioned earlier, this formulation just gives the shape of the posterior distribution, leaving us incapable of doing any inference about the parameter vector $\boldsymbol{\theta}$. To find the actual posterior, we need to calculate the scaling factor of Eq. (2.8) which may be very difficult to solve.

Deterministic approximation methods allow the posterior to be approached from a completely different direction. These are based on analytical approximations to the

posterior distribution, by assuming that they have some properties or have a specific parametric form. As such, they transform the problem of integration to the task of finding the solutions to a mathematical optimization problem. Here we will give a brief overview of a family of deterministic approaches called *variational Bayes* techniques which have been widely applied in practice. Some other deterministic approximation methods, such as *expectation propagation* [Minka, 2001a,b], optimize different forms of the same objective function that we will explain below.

Variational Inference

In general, variational methods [Wainwright & Jordan, 2008; Beal, 2003; MacKay, 2003] aim to approximate a complex posterior distribution using a distribution that has a simpler structure, typically referred to as a *variational distribution*. Suppose we adopt a fully Bayesian approach in which our parameters are given prior distributions that may belong to some parametric family. Therefore, since our original parameter vector is now a vector of random variables, it is preferable to denote the set of all latent variables and parameters by $\mathbf{z}$.

The dissimilarity between the variational distribution $q(\mathbf{z})$ and the posterior distribution $p(\mathbf{z}|\mathbf{y})$ is measured in terms of *Kullback-Leibler* (KL) divergence. Let (Θ, Σ) be the σ-algebra over which our random variable $\mathbf{z}$ is defined. Then by definition, the KL divergence between the variational distribution and the true posterior is the integral:

$$\mathrm{KL}(q||p) = \int_{\Theta} q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z}|\mathbf{y})} d\mathbf{z} \tag{2.40}$$

The KL-divergence has the property of being zero if the two distributions are equal, and positive otherwise. The goal is then to find the distribution, or the settings of parameters, that minimize the KL divergence with respect to q[1]. Now elaborating on

[1]It should be noted that the KL-divergence is not symmetric, i.e. $KL(q||p)$ is not necessarily the same as $KL(p||q)$. This distinction is important because the distribution which minimizes the former is not the same as the one which minimizes the latter (See e.g. Bishop [2006] or Winn [2004] for more details)

Eq. (2.40), replacing $p(\mathbf{z}|\mathbf{y})$ with its equivalent $p(\mathbf{z},\mathbf{y})/p(\mathbf{y})$, we get:

$$\begin{aligned}
\mathrm{KL}(q||p) &= \int_\Theta q(\mathbf{z}) \log \frac{q(\mathbf{z})p(\mathbf{y})}{p(\mathbf{z},\mathbf{y})} d\mathbf{z} \\
&= \int_\Theta q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z},\mathbf{y})} d\mathbf{z} + \int_\Theta q(\mathbf{z}) \log p(\mathbf{y}) d\mathbf{z} \\
&= -\mathcal{L}(q) + \log p(\mathbf{y}) \int_\Theta q(\mathbf{z}) d\mathbf{z} \\
&= -\mathcal{L}(q) + \log p(\mathbf{y}) \qquad (2.41)
\end{aligned}$$

and $\mathcal{L}(q)$ is by definition equal to:

$$\mathcal{L}(q) = \int_\Theta q(\mathbf{z}) \log p(\mathbf{z},\mathbf{y}) d\mathbf{z} - \int_\Theta q(\mathbf{z}) \log q(\mathbf{z}) d\mathbf{z} \qquad (2.42)$$

The second term in Eq. (2.42) is the negative of the entropy of q, denoted by $H(q)$, which measures the amount of uncertainty in our variational distribution. Also, the first term is the expectation of $\log p(\mathbf{z},\mathbf{y})$ with regard to $q(\mathbf{z})$. This amounts to the following simplified equality:

$$\mathcal{L}(q) = E_{q(\mathbf{z})} \left[\log p(\mathbf{z},\mathbf{y})\right] + H(q) \qquad (2.43)$$

Note that the last term in Eq. (2.41) can be ignored since it does not depend on q. Hence, we should chose an appropriate form of distribution for q whose entropy as well as its expectation in Eq. (2.43) are calculable. We can then maximise the value of $\mathcal{L}(q)$ using standard optimization methods, so as to minimize the KL-divergence.

One common way to simplify the variational distribution is to choose q such that disjoint groups of variables are independent, also called factorized approximation [Winn, 2004; Bishop, 2006]. Properties of such factorized distributions, along with examples of factorized variational approximation for members of the exponential family can be found in MacKay [2003].

While variational methods scale well in large applications, they can never generate exact results. Stochastic approximation methods, on the other hand, overcome this weakness of variational approaches by tackling the problem from another point of view; instead of trying to approximate the true posterior with another distribution, these methods rely on numerical samples from the posterior. Even though stochastic approaches are computationally demanding, but they have the advantage that with enough resources, they can achieve any desired level of accuracy.

2.4.2 Stochastic Approximation

Stochastic approximation techniques, also called Monte Carlo methods, aim to replace the difficult numerical integration with the much easier task of drawing random samples. Like many other mathematical breakthroughs, they were mainly developed by physicists in their attempt to use random number generation to compute complex integrals [Meteopolis & Ulam, 1949].

Monte Carlo Integration

To see the motivation behind Monte Carlo approaches, consider computing a complex integral of the form:

$$\int_{\Theta} h(x)dx \tag{2.44}$$

If we are able to decompose the function $h(x)$ into the product of a function $f(x)$ and a probability density function $p(x)$ over a σ-algebra Σ, then the integral turns into:

$$\int_{\Theta} h(x)dx = \int_{\Theta} f(x)p(x)dx = E_{p(x)}\left[f(x)\right] \tag{2.45}$$

Now if we draw a large number of samples $X = \{x_1, x_2, \ldots, x_n\}$ from the density $p(x)$, and calculate their empirical average by:

$$\bar{f}_n = \frac{1}{n}\sum_{i=1}^{n} f(x_i) \tag{2.46}$$

then by the Strong Law of Large Numbers[1], $\bar{f}_n$ almost surely[2] converges to the expectation $E_{p(x)}\left[f(x)\right]$. Hence we can approximate the integral of Eq. (2.44) using random numbers generated from the density function $p(x)$.

This approach, also called *Monte Carlo integration* [Robert & Casella, 2004; Walsh, 2004], can be extensively used in Bayesian statistics since most calculations require some sort of integration. For instance, the marginal distribution as defined in Eq. (2.8), for a single parameter can be approximated by:

[1]The strong Law of Large Numbers states that the sample average converges almost surely to the expected value. For a proof, see e.g. Loève [1977]

[2]Let (Θ, Σ, P) be a probability space, then event $e \in \Sigma$ happens almost surely if $P(e) = 1$

$$p(\mathbf{y}) = \int_{\Theta} p(\theta)p(\mathbf{y}|\theta)d\theta \simeq \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{y}|\theta_i) \tag{2.47}$$

where $\{\theta_1, \theta_2, \ldots, \theta_n\}$ are samples drawn from our prior. However, we should still address the problem of how to generate random samples from a given function.

Direct Sampling Algorithms

Many methods have been devised to draw random samples from a function. These methods may often assume that we have access to random numbers from a uniform distribution; i.e. we can generate random numbers between 0 and 1.

In *Importance Sampling*, we find a proper density $q(x)$ which roughly estimates our target distribution $p(x)$ and could easily be sampled (The method can work with any choice of the distribution $q(x)$ as long as the support[1] of p is a subset of the support of q [Robert & Casella, 2004]). We then rewrite Eq. (2.45) as:

$$\int_{\Theta} f(x)p(x)dx = \int_{\Theta} f(x)\left(\frac{p(x)}{q(x)}\right)q(x)dx \simeq \frac{1}{n}\sum_{i=1}^{n} f(x_i)\left(\frac{p(x_i)}{q(x_i)}\right) \tag{2.48}$$

where the samples $\{x_1, x_2, \ldots, x_n\}$ are drawn from the proposal distribution $q(x)$.

Rejection Sampling is another direct sampling technique in which an easy-to-sample envelope distribution is used for drawing our samples. However, the proposal distribution must satisfy:

$$p(x) \leq Mq(x) \qquad \text{where} \qquad M < \infty \tag{2.49}$$

The algorithm then draws samples from the distribution $q(x)$ and for each sample x_i, accepts it with probability $r(x_i) = \dfrac{p(x_i)}{Mq(x_i)}$ and rejects it otherwise. This is done by generating a random number u between 0 and 1 and verifying whether $r(x_i)$ is greater/less than u in order to accept/reject the sample.

While there is a rich literature on direct sampling algorithms, we will not go into their details any further. Interested readers may refer to Bolstad [2010] or Bishop [2006] for more explanation. Nevertheless, it should be noted that it is often impossible to find an easy to sample distribution which is also a good approximation at the same

[1] The support of a probability measure P is any σ-algebra $\Sigma \in \Theta$ for which $P(\Sigma) = 1$

time [Andrieu et al., 2003]. In the next section, we introduce a very general and powerful framework of stochastic approximation methods that overcome some of these limitations.

Markov Chain Monte Carlo Algorithms

Markov Chain Monte Carlo [Andrieu et al., 2003; Chen et al., 2000; Robert & Casella, 2004] or MCMC is a general method based on drawing sequential samples from approximate distributions and then correcting those draws to better approximate the target distribution. The goal is to set up a chain such that the desired target distribution p is its equilibrium distribution. Thus before discussing these algorithms, it is insightful to give a few introductory comments on these so called *Markov Chains* in order to help better understand the techniques presented later.

Markov Chains

Let $\mathbf{z}^t$ denote the value of a vector of random variables at time t. Then the sequence of such random variable vectors is called a *Markov Chain* [Lefebvre, 2006; Bishop, 2006] if the following conditional independence property holds:

$$p(\mathbf{z}^{t+1}|\mathbf{z}^1, \ldots, \mathbf{z}^t) = p(\mathbf{z}^{t+1}|\mathbf{z}^t) \tag{2.50}$$

This equation is also called *Markov's property*. It simply states that the future, given the past and the present, is independent of the past and only depends on the present [Lefebvre, 2006]. The Markov chain is basically a *Stochastic Process*, a concept that we will deal with extensively in the next chapters.

We can specify a Markov chain by giving the probability of the starting vector $p(\mathbf{z}^0)$ along with the *transition probabilities* for subsequent states in the chain; i.e. $p(\mathbf{z}^{t+1}|\mathbf{z}^t)$. The chain is *homogeneous* if the transition probabilities remain invariant for all t.

A distribution p^* is called *invariant* if:

$$p^*(\mathbf{z}) = \sum_{\mathbf{z}'} p(\mathbf{z}'|\mathbf{z})p^*(\mathbf{z}') \tag{2.51}$$

It can be shown that subject to weak restrictions, a homogeneous Markov chain will converge to an invariant distribution irrespective of the choice of initial distribution $p(\mathbf{z}^0)$ [Neal,

1993]. This property is called *ergodicity* and the invariant distribution is called the *equilibrium* distribution.

Markov Chain Monte Carlo Sampling

MCMC works like rejection and importance sampling in the sense that the samples are drawn from some distribution other than the original target distribution. But this time, the distribution does not generate i.i.d samples; at each step, we record the current state $\mathbf{z}^t$, and the proposal distribution $q(\mathbf{z}^{t+1}|\mathbf{z}^t)$ draws the next sample based on the the current state. Therefore, the sequence of samples $\{\mathbf{z}^1, \mathbf{z}^2, \dots \mathbf{z}^n\}$ forms a Markov chain.

The *Metropolis-Hastings* algorithm [Metropolis et al., 1953; Hastings, 1970] is the most popular MCMC method. Given the current value of our sample, the Markov chain accepts the new candidate value $\mathbf{z}^{t+1}$ as its new state with probability α equal to:

$$\alpha = \min\left(1, \frac{p(\mathbf{z}^t)q(\mathbf{z}^{t+1}|\mathbf{z}^t)}{p(\mathbf{z}^{t+1})q(\mathbf{z}^t|\mathbf{z}^{t+1})}\right) \tag{2.52}$$

otherwise the chain remains in its current state. It can be shown (see, e.g. Robert & Casella [2004]) that the target distribution p is indeed the equilibrium distribution of the Markov chain defined above.

Gibbs Sampling

Gibbs sampling [Casella & George, 1992], probably the best known MCMC sampling algorithm in Bayesian statistics, is a special case of the Metropolis-Hastings algorithm where we always accept the value of the drawn sample. Suppose $\mathbf{z} = (z_1, z_2, \dots, z_k)$ is our parameter vector and our goal is to sample from the target distribution $p(\mathbf{z})$. The basic algorithm for this sampling scheme is given below:

An important feature of the Gibbs sampling algorithm is that all we need for our simulation is the conditional probabilities shown in Algorithm 1. Therefore, even when we are dealing with high dimensional spaces, all of our simulations are univariate [Robert & Casella, 2004].

There are several variants to the standard Gibbs sampling algorithm which try to improve its performance. In *grouped Gibbs sampling*, also called *blocked Gibbs sampling*,

Algorithm 1: Gibbs sampling from the distribution $p(z_1, z_2, \ldots, z_k)$

Choose a starting point $\mathbf{z}^0 = (\mathbf{z}_1^0, \mathbf{z}_2^0, \ldots, \mathbf{z}_k^0)$;
Generate new samples as follows
for $i + 1 = 1$ **to** T **do**
    Sample $z_1^{i+1} \sim p(z_1^i | z_2^i, z_3^i, \ldots, z_k^i)$;
    Sample $z_2^{i+1} \sim p(z_2^i | z_1^i, z_3^i, \ldots, z_k^i)$;
    $\vdots$
    Sample $z_j^{i+1} \sim p(z_j^i | z_1^i, \ldots, z_{j-1}^i, z_{j+1}^i, \ldots, z_k^i)$;
    $\vdots$
    Sample $z_k^{i+1} \sim p(z_k^i | z_1^i, z_2^i, \ldots, z_{k-1}^i)$;
end

we form blocks of two or more variables together and then sample from their joint distribution conditioned on all other variables. In some cases, we may even be able to analytically marginalize over some of the variables, which is often referred to as *collapsed Gibbs sampling*. See Chen et al. [2000] or Fox [2009] for a more detailed analysis of different variants of Gibbs sampling, and MCMC techniques in general.

2.5 Summary

This chapter dealt with some of the most important mathematical concepts that will be used throughout the thesis. We introduced the basics of parameter estimation and prediction in the Bayesian framework, and reviewed the Dirichlet distribution which is at the heart of all the concepts provided hereafter. We also covered inference in Bayesian statistics which is of utmost importance when dealing with integrals that do not have analytic solutions. Among these approaches, Gibbs sampling is perhaps the most commonly used numerical method for approximating complex integrals. In the next chapter, we will use these concepts and discuss nonparametric Bayesian methods, which overcome the limitations of traditional parametric methods.

Chapter 3

Nonparametric Bayesian Analysis

Whereas parametric Bayesian methods comprise models with a finite (and often low) number of parameters, nonparametric Bayesian approaches [Ferguson, 1973; Ghosh & Ramamoorthi, 2003; Hjort et al., 2010] avoid the use of restricted functional forms and thus are not limited to finite parametrizations. Strictly speaking, nonparametric here does not mean a parameter-less model, rather a model in which the number of parameters grow as more data are observed. Nonparametric Bayesian models have gained remarkable popularity in the field of machine learning because of their flexibility, especially in unsupervised learning. In the following sections, we describe one of the cornerstone classes of Bayesian nonparametrics: the Dirichlet process, its representation, and infinite dimensional mixture models which are based on it.

3.1 Stochastic Processes

Stochastic Processes [Billingsley, 1995; Lefebvre, 2006] are the building blocks that make nonparametric Bayesian methods capable of dealing with infinite dimensional parameter spaces. Loosely speaking, a stochastic process is a process that stochastically evolves through time, in the sense that if we repeat the experiment under the same conditions, the states that we move through are different than the states we observed during the first run. A more formal definition is given below:

Definition 3.1. A stochastic process is a collection of random variables $\{X(t) : t \in T\}$ indexed by a set T (which can be though of as representing time) on a probability space (Θ, Σ, P).

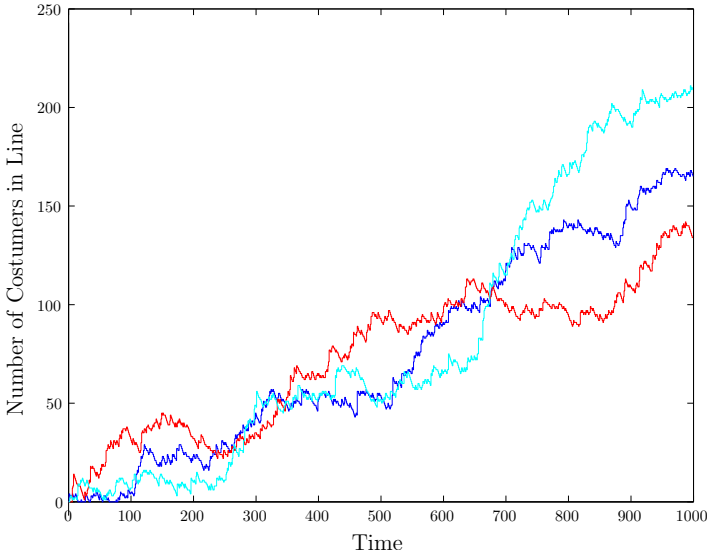

Figure 3.1: A stochastic process where people randomly enter a waiting line to get some kind of service. Different colours show separate executions of the process

As an example, consider a waiting line of customers needing some kind of service where new costumers can enter the line at discrete time units. Only one costumer can get service in every time unit. Let $s(k)$ denote the number of customers joining the line at time k, then the size of the line at time $k > 1$, $l(k)$, is given by

$$l(k) = \begin{cases} l(k-1) + s(k) & \text{if the line is empty,} \\ l(k-1) + s(k) - 1 & \text{otherwise.} \end{cases}$$

where we subtract 1 in the second line to indicate the customer that is getting his service at time k. This process satisfies the Markov property described in Eq. 2.50, because the value of the next state depends only on the current state of the system, and is therefore an instance of a Markov chain. Figure 3.1 shows three different executions of the process, under the same conditions, where the arrival time of costumers is modelled by a binomial distribution. As can be seen in the figure, each run returns a different trajectory of state transitions.

Stochastic processes are often identified by the specific distribution they induce on their finite statistics. For example, the *Gaussian process* [Rasmussen & Williams, 2006; MacKay, 1998] provides a distribution over real valued functions such that any finite subset of the range follows a *multivariate Gaussian distribution*. In the following sections, we describe the Dirichlet process due to their significant role in nonparametric Bayesian modelling, and elaborate on some of their important properties.

3.2 The Dirichlet Process

The *Dirichlet process* [Teh, 2010; Sudderth, 2006; Fox, 2009; Rodrigez, 2007] over a set S is a stochastic process where every trajectory is a probability distribution over partitions of S. We can think of the Dirichlet process as an extension of the Dirichlet distribution to infinite dimensional spaces, such that every finite group of data, regardless of the grouping mechanism, follows a Dirichlet distribution. In other words, for any finite measurable partition[1] $\{A_1, A_2, \dots A_k\}$ of S, the probability vector $(p(A_1), p(A_2), \dots, p(A_K))$ must be a k-dimensional Dirichlet distribution. However, this is a very delicate condition and not so easy to satisfy. To see the reason, consider breaking one of the A_i's into a measurable partition $\{B_1, B_2, \dots, B_m\}$. Then by the agglomeration property of Dirichlet distributions, as explained in Theorem 2.2, by adding up the Dirichlet parameters corresponding to the new partition, we must obtain the parameter of the orginial A_i that we partitioned. It thus follows that in order to reach our goal, we must assume the parameter set to be constrained by countable additivity. Therefore, a wise choice would be to consider the parameters of the Dirichlet distribution to actually be measures themselves because countable additivity is fused into what we expect from a measure (see Definition 2.2).

It can be shown that the measure we are assuming for the parameter set could be uniquely decomposed into a *base probability measure* H and a *concentration parameter* γ. The whole argument can be summarized in the following Theorem:

Theorem 3.1. *Let H be a probability distribution on a measurable space (Θ, Σ) and γ a positive scalar. A probability measure G on Θ is called a Dirichlet process (DP) if every finite measurable partition $\{A_1, A_2, \dots, A_k\}$ on Θ follows a k-dimensional Dirichlet distribution:*

$$p((G(A_1), G(A_2), \dots, G(A_k))|\gamma, H) \sim Dirichlet(\gamma H(A_1), \gamma H(A_2) \dots, \gamma H(A_k)) \quad (3.1)$$

For any base measure H and concentration parameter γ, there is a unique stochastic process satisfying these conditions, which we denote by $DP(\gamma, H)$.

This theorem was originally proved by Ferguson [1973] in his influential work, and later by Sethuraman [1994] using a more constructive definition which we will describe in Section 3.2.2.

[1] Let (Θ, Σ) be a measurable space, we call $\{A_1, A_2, \dots, A_k\}$ a measurable partition of Θ if $A_i \in \Sigma$ for all i, $A_i \cap A_j = \emptyset$ for $i \neq j$ and $\cup_{i=1}^{k} A_i = \Theta$

The parameters H and γ play significant roles in understanding the Dirichlet Process. To see their relative importance, consider the expected value and variance of a DP for any region $A \in \Theta$ by combining Eqs. (2.32) and (3.1):

$$E(G(A)) = H(A) \qquad Var(G(A)) = \frac{H(A)(1 - H(A))}{\gamma + 1} \tag{3.2}$$

Therefore, the base distribution H is the mean value of our DP and the concentration parameter γ can be understood as an inverse variance which determines the average deviation of the samples from the base measure.

Figure 3.2 shows how different settings of the concentration parameter affect the samples drawn from a specific representation of the Dirichlet Process referred to as the *stick-breaking construction*. We will thoroughly explain this representation in Section 3.2.2. For the moment, just observe that as $\gamma \to \infty$, the samples will be more concentrated around the mean and $G(A) \to H(A)$. However, care must be taken to avoid confusion with $G \to H$ since the former does not imply the latter. See Teh [2010] for more details.

3.2.1 Posterior Distribution

Assume that we draw a sample G from a Dirichlet process $\text{DP}(\gamma, H)$. Since G is itself a probability distribution, we can in turn draw samples from G. Let $\mathbf{y} = (y_1, \ldots, y_n)$ be the observations that we have made from the distribution G, then for any finite measurable partition $\{A_1, A_2, \ldots, A_k\}$ on Θ, the posterior distribution of G given our sequence of observations is given by:

$$p((G(A_1), \ldots, G(A_k))|y_1, \ldots, y_n) \sim \text{Dirichlet}(\gamma H(A_1) + n_1, \ldots, \gamma H(A_k) + n_k) \tag{3.3}$$

where n_i indicates how many of our observations fall into partition A_i. As shown by Ferguson [1973], this result is due to the fact that every finite partition of the DP follows a Dirichlet distribution and by the Dirichlet-Multinomial conjugacy described in Eq. (2.35), we just have to update the parameters of our distribution with the number of observations in each corresponding partition. See Sethuraman [1994] for an alternative proof.

Note that with a little algebra, we can see that $n_i = \sum_{j=1}^{n} \delta_{\theta_j}(A_i)$ where the fucntion δ is the the Kronecker delta function explained in Section 2.3.2. Since Eq. (3.3) is true

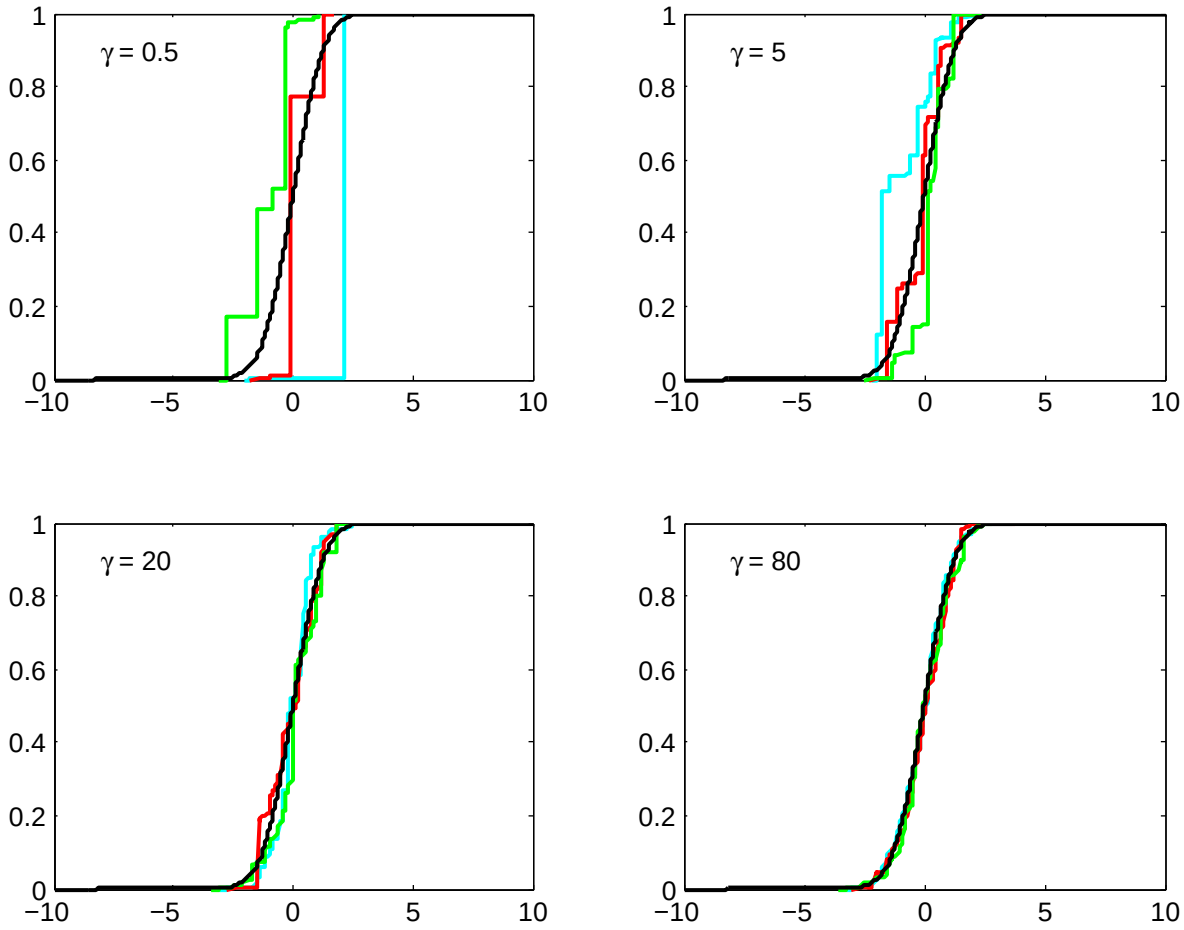

Figure 3.2: Samples of a Dirichlet Process for different concentration parameters with a standard Gaussian base measure. The cumulative distribution function of the Gaussian measure is drawn in black. As the value of γ increases, the results tend to be closer to the baseline measure. In all cases, the samples are centred around the base distribution.

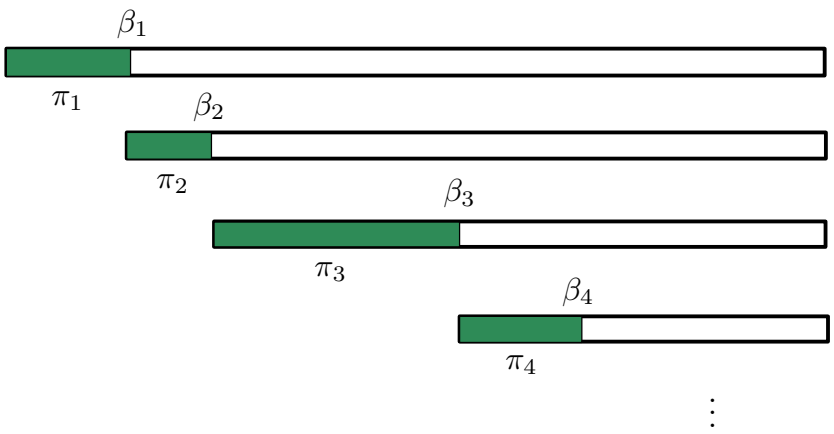

Figure 3.3: The stick-breaking construction of the Dirichlet process. The construction works by sequentially breaking a unit-length stick at random points obtained from Beta$(1, \gamma)$.

for every finite measurable partition, we can formalize this analysis and rewrite the posterior as:

$$G|y_1, \ldots, y_n, H, \gamma \sim \mathrm{DP}(\gamma + n, \frac{\gamma}{\gamma + n} H + \frac{1}{\gamma + n} \sum_{i=1}^{n} \delta_{y_i}) \tag{3.4}$$

Equation (3.4) shows that the relative weight of the prior base distribution is proportional to γ. Thus we can characterize γ as the strength associated with our prior H. As we will see in Section 3.2.3, the base distribution H also plays a role in the predictive distribution for the Dirichlet process.

We have already observed many characteristic of the Dirichlet process, but none of these provide an approach to make predictions or to draw samples from the DP in order to make inferences. In the following sections, we will introduce some important representations of the Dirichlet process where each unveils significant properties of the DP. Many of the applications of the Dirichlet process are based on these more intuitive representations which also play an important role from a computational point of view. Each representation is based on an important property of the DP that we will describe in detail.

3.2.2 Discreteness and the Stick-Breaking Construction

The *Stick-breaking construction* [Sethuraman, 1994; Sudderth, 2006] is an important representation of the Dirichlet process which stems from a very important characteristic

of the DP. In the previous section, we showed that the posterior distribution for any finite measurable partition is also a Dirichlet process with parameters updated as in Eq. (3.4). Using this result in conjunction with Eq. (3.2), we see that for any $A \in \Theta$:

$$E\left(G(A)|y_1, \ldots, y_n, H, \gamma\right) = \frac{\gamma}{\gamma + n} H + \frac{n}{\gamma + n} \frac{\sum_{i=1}^{n} \delta_{y_i}}{n} \tag{3.5}$$

Notice that distributions of the form $pR + (1-p)S$, where R and S are distributions, represent the distribution that is a mixture of R and S with mixing proportions p and $1-p$ respectively. Now if we take the limit as the number of observations approaches infinity, we get [Fox, 2009; Sudderth, 2006]:

$$\lim_{n \to \infty} E\left(G(A)|y_1, \ldots, y_n, H, \gamma\right) = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k^*} \tag{3.6}$$

where the vector $\boldsymbol{\theta}^* = (\theta_1^*, \theta_2^*, \ldots)$ contains the distinct values among our set of observations, and the $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots)$ represents their corresponding empirical frequency. Bearing in mind that draws from the DP tend to concentrate around their mean, based on Eq. (3.6) it is natural to think of the sample distributions to be discrete probability measures. The following theorem gives a formal account of this result, along with a straightforward construction of the Dirichlet process based on this important property of the DP.

Theorem 3.2. *Let $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots)$ be and infinite sequence of weights derived from the following stick-breaking process with parameter $\gamma > 0$:*

$$\beta_k \sim Beta(1, \gamma), \qquad i = 1, 2, \ldots \tag{3.7}$$

$$\pi_k = \beta_k \prod_{j=1}^{k-1} (1 - \beta_j) \tag{3.8}$$

Given a base measure H on Θ, suppose we draw an infinite number of samples $\theta_k^* \sim H$ and form the following discrete probability measure:

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k^*} \tag{3.9}$$

Then this construction returns a Dirichlet process with concentration parameter γ and base distribution H, i.e. $G \sim DP(\gamma, H)$. On the other hand, all samples from a Dirichlet process are almost surely discrete and have a representation as in Eq. (3.9).

The almost sure discreteness of the Dirichlet process was first proven by Ferguson

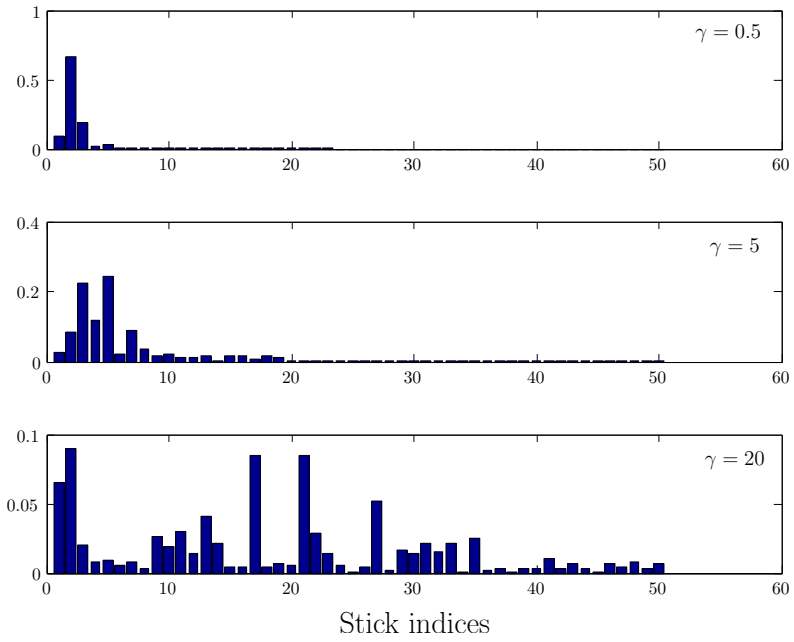

Figure 3.4: Three random stick-breaking constructions with different values of γ. Each bar represents the length of the stick at than index. For small values of the concentration parameter, the probability mass is mostly focused on the initial sample points, whereas increasing γ makes the process more diffuse.

[1973]. Later, Sethuraman [1994] used this property to prove that the construction above actually represents the Dirichlet process.

To see where the terminology of stick-breaking comes from, suppose we have a unit-length stick and break it at position β_1. Let π_1 be equal to the length of the broken stick which is β_1 in this case. We then break the remainder of the stick at β_2 and let π_2 be equal to the length of the new broken part. This process is continued until we obtain our sequence of π_i's. Figure 3.3 graphically demonstrates how this process works. Having built our measure as in Eq. (3.9), Theorem 3.2 assures us that this construction always returns a Dirichlet process. The stick-breaking process is usually denoted by $\pi \sim \text{GEM}(\gamma)$, where GEM comes from the first letters of Griffiths, Engen and McCloskey [Pitman, 2006].

The stick-breaking construction can be seen as a special representation of a more general case where the sample weights are drawn from $\text{Beta}(a_k, b_k)$. For example, considering $a_k = a$ and $b_k = b$ returns the *Beta two parameter process* [Ishwaran & Zarepour, 2000a]. This stochastic process can be extended to obtain the Poisson-Dirichlet process [Pitman & Yor, 1997], also called the *Pitman-Yor process*, where the two parameters are indexed by $a_k = 1 - a$ and $b_k = b + ka$. The Pitman-Yor process gives more flexibility

than the Dirichlet process and has been proven to be more suited to applications where we model data with power-law tails (e.g., word frequencies in natural language) [Teh, 2006; Goldwater et al., 2006]. As noted by Pitman and Yor themselves, letting $a = 0$ and $b = \gamma$ returns the Dirichlet process which we covered in detail in this section. In the next section, we show how this construction can lead to the predictive distribution of the Dirichlet process.

3.2.3 Prediction and the Pólya Urn Scheme

Suppose we draw a discrete measure $G \sim \text{DP}(\gamma, H)$, and draw the i.i.d samples $(y_1, \dots y_n) \sim G$. We can then derive the predictive distribution for y_{n+1} based on Eq. (2.12) by integrating G out:

$$p(y_{n+1}|y_1, \dots, y_n, H, \gamma) = \int_\Theta p(y_{n+1}|y_1, \dots, y_n, G, \gamma, H) p(G|y_1, \dots, y_n, H, \gamma) dG \tag{3.10}$$

Considering the fact that the samples are conditionally independent given the measure G, Eq. (3.10) can be further simplified as:

$$\begin{aligned} p(y_{n+1}|y_1, \dots, y_n, H, \gamma) & \\ &= \int_\Theta p(y_{n+1}|G, \gamma, H) p(G|y_1, \dots, y_n, H, \gamma) dG && (3.11) \\ &= \int_\Theta G p(G|y_1, \dots, y_n, H, \gamma) dG && (3.12) \\ &= E(G|y_1, \dots, y_n, H, \gamma) && (3.13) \end{aligned}$$

where Eq. (3.12) follows from the fact that y_{n+1} is drawn from the discrete measure G. We can now formally characterize the predictive distribution for the next observation y_{n+1} as follows:

Theorem 3.3. *Let $G \sim DP(\gamma, H)$ be distributed according to a Dirichlet process, and consider a set of n observations $(y_1, \dots y_n) \sim G$ taking $K \leq n$ distinct values $\boldsymbol{\theta}^* = (\theta_1^*, \dots, \theta_K^*)$. The predictive distribution of the next observation is equal to:*

$$p(y_{n+1}|y_1, \dots, y_n, H, \gamma) = \frac{1}{\gamma + n} \left(\gamma H + \sum_{k=1}^{K} n_k \delta_{\theta_k^*} \right) \tag{3.14}$$

where n_k is equal to the number of observations equal to θ_k^*

In our discussion on the stick-breaking construction in Section 3.2.2, we saw that draws from a Dirichlet process are discrete with probability one. Hence there is always a positive probability of multiple observations taking the same value. In other words, n observations $\mathbf{y} = (y_1, \dots y_n) \sim G$ will always take $K \leq n$ values $\boldsymbol{\theta^*} = (\theta_1^*, \dots, \theta_K^*)$. The rest of the proof follows directly from replacing Eq. (3.13) by Eq. (3.5). For a more formal argument, see Blackwell & MacQueen [1973].

As with many probability distributions which can be described using urn models [Mahmoud, 2009], the model that corresponds to the Dirichlet distribution is called the *Pólya urn scheme* [Teh, 2010; Fox, 2009]. Accordingly, Blackwell & MacQueen [1973] proposed a generative process based on Theorem 3.3 to represent the Dirichlet process. Suppose we choose a ball with any random color and place it in an initially empty urn. Then in each of the consequent steps, we either pick a ball at random from the urn and return it along with an additional ball of the same color θ^*, or we place a special ball with a new unseen color in the urn. The first case, which involves drawing an existing ball from the urn, is chosen with probability proportional to n (where n is the number of balls already in the urn) and the second case where we chose a new ball, is chosen with probability proportional to γ normal balls. This process allows us to directly draw samples from a Dirichlet process without actually constructing the underlying DP.

3.2.4 Clustering and the Chinese Restaurant Process

From Theorem 3.3, we know that samples from the Dirichlet processes are discrete probability measures made up of a countably infinite number of point masses. Therefore, there is always a non-zero probability of two samples colliding. This implies a clustering property for the Dirichlet process where the samples are partitioned into a finite number of clusters.

In our discussion on the posterior distribution of the Dirichlet process, we showed that n observations $\mathbf{y} = (y_1, \dots y_n) \sim G$ take $K \leq n$ distinct values $\boldsymbol{\theta^*} = (\theta_1^*, \dots, \theta_K^*)$. This means that we have clustered our data into K distinct clusters. Let $\mathbf{z} = (z_1, z_2, \dots, z_n)$ be the vector of indicator variables that assigns each observation to it's corresponding cluster; i.e.:

$$y_i = \theta_{z_i}^*, \qquad 1 \leq z_i \leq K \tag{3.15}$$

Now if we rewrite the predictive distribution for the indicator variables, Eq. (3.14) shows that:

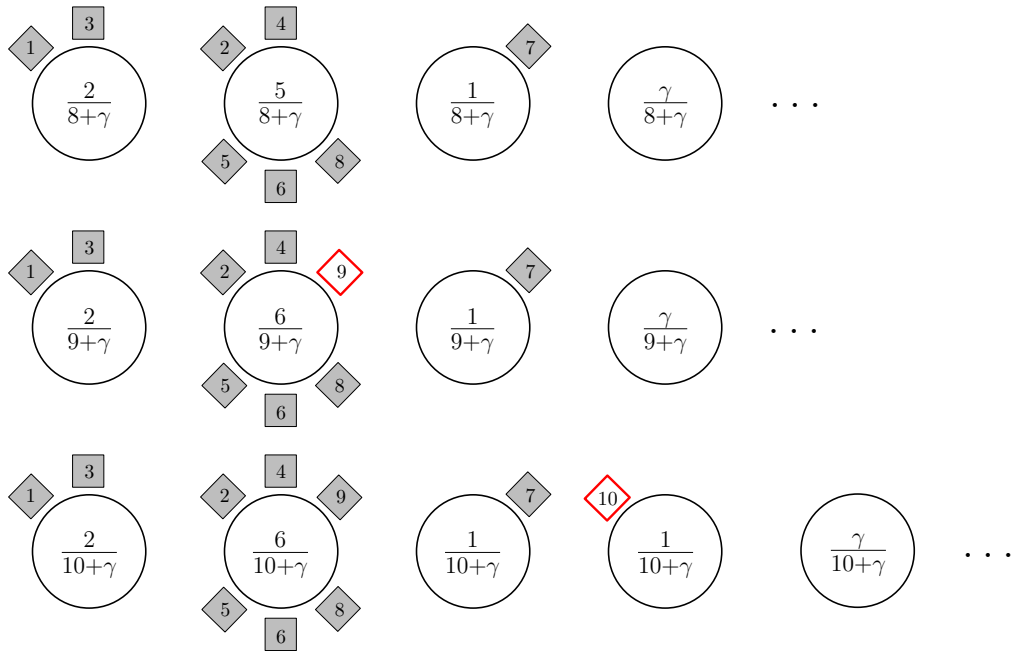

Figure 3.5: Demonstration of three consecutive steps of the Chinese restaurant process with parameter γ. The circles are analogous to tables (clusters) and the rectangles represent the customers (observations). Each table is labelled by the probability that a new customer sits there.

$$p(z_{n+1}|z_1,\ldots,z_n,\gamma) = \frac{1}{\gamma+n}\left(\gamma\delta_{K+1} + \sum_{k=1}^{K} n_k\delta_k\right) \tag{3.16}$$

where n_k is the number of observations in cluster k, and $K+1$ is a new unseen cluster.

This distribution on partitions in Eq. (3.16) is commonly referred to as the *Chinese Restaurant Process* (CRP) [Pitman, 2006; Aldous, 1985]. The analogy, which stems from the fact that there are large number of tables in a Chinese restaurant, is as follows. Consider a Chinese restaurant with an infinite number of tables, each of which can seat an infinite number of customers. The first customer enters and sits at the first table. In each subsequent step, as the n'th customer enters, she sits at an already occupied table $1 \leq k \leq K$ with probability proportional to the number of people already sitting there; i.e. $\frac{n_k}{\gamma+n}$, or chooses a new table $K+1$ with probability proportional to $\frac{\gamma}{\gamma+n}$. Fig. 3.5 gives a graphical representation of how this process works. The fact that most Chinese restaurants have round tables is an important aspect of the CRP. This is because a Dirichlet process is not only a distribution over partitions, it also defines a distribution over their permutations.

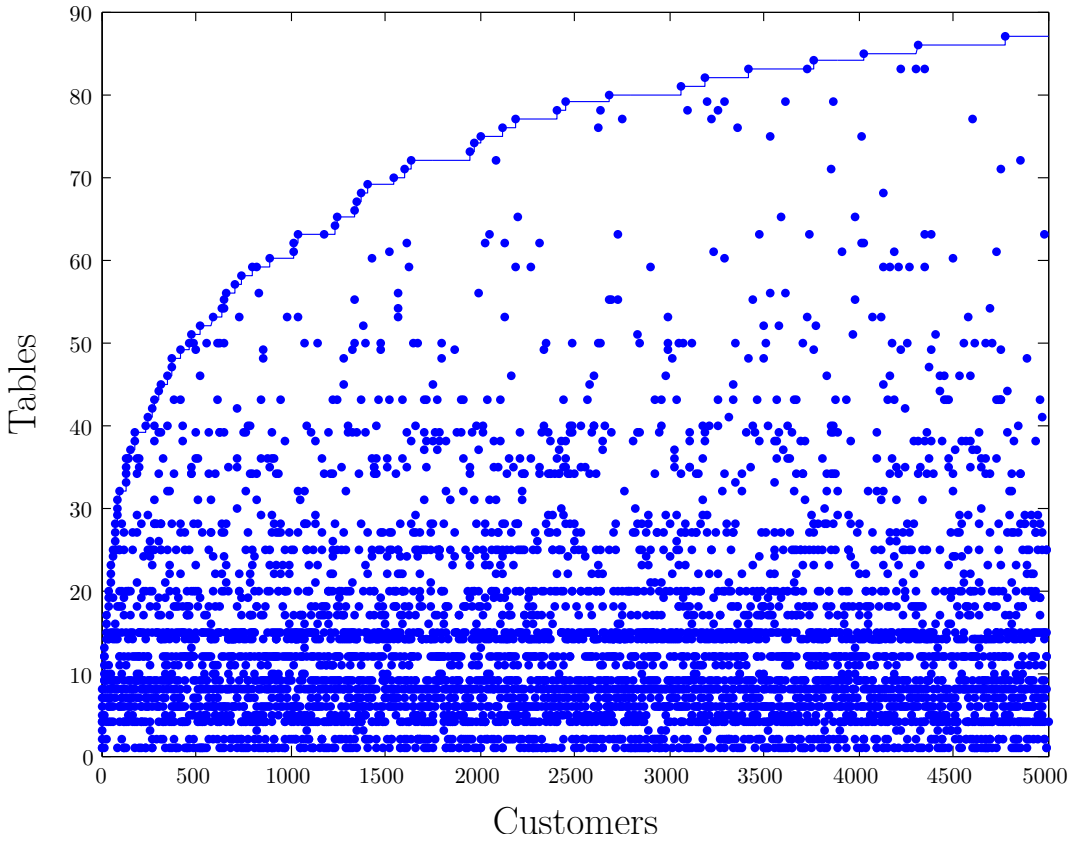

Figure 3.6: A sample assignment of observations to clusters in a Chinese restaurant process with parameter $\gamma = 15$. As the number of observations grows, the number of clusters grows only logarithmically in the number of observations.

This construction also exhibits a phenomenon that is often referred to as the rich-gets-richer phenomenon [Teh, 2010]. Observe that in Eq. (3.16), a customer chooses an already occupied table based on the number of people who are already sitting there. Therefore, large tables (tables for which n_k is large) grow even larger. Figure 3.6 shows how newly entered customers tend to choose the first few tables, which are naturally crowded. This property suggests that even if in principle we can have an infinite number of tables, the actual number of occupied tables will be much less than the total number of customers. In Section 3.2.5 we will show that this number grows only logarithmically in the number of observations.

As we will see in the rest of this thesis, the Chinese restaurant process is widely used in nonparametric Bayesian analysis because they let us define statistical models where the observed data are allowed to be drawn from an unknown number of clusters. There are also many variations to the CRP some of which are briefly described below.

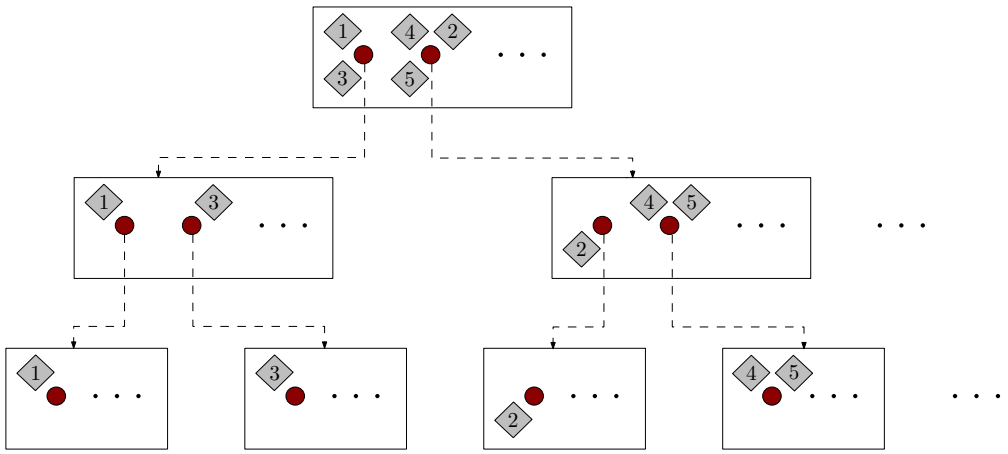

Figure 3.7: Demonstration of five sample paths of the nested Chinese restaurant process for a three level nCRP. Here, five customers have entered the root restaurant that has a card referring to another restaurant for the next day on each of its infinite tables. The same structure repeats for each of the restaurants on the subsequent levels.

The nested Chinese Restaurant Process

The *nested Chinese Restaurant Process* (nCRP) [Blei et al., 2004] assumes that we have an infinite number of infinite-table Chinese restaurants. As demonstrated in Fig. 3.7, this is achieved by considering an infinitely deep tree structure with each level having an infinite number of branches.

To see how the nCRP works, consider a tourist which has decided to taste the foods of a different restaurant every day on her L day trip. On the first evening, she enters the root restaurant and chooses a table based on Eq. (3.16). On each table, there is a card suggesting another restaurant (each restaurant is referred to exactly once); so she picks up the card and visits the suggested restaurant on her next day. Again, she chooses a table from that restaurant, and continues this process for the L days she is visiting there. At the end, she has formed a path of length L from the root restaurant to the L-level tree described above.

While the CRP can be used for modelling uncertainty about the number of clusters, the nested Chinese restaurant process can be used to model nested tree topologies. This feature makes them excellent candidates for hierarchical topic modelling applications where as we go deeper into the tree, the corresponding topics at each level will be more specialized to that particular document [Blei et al., 2010].

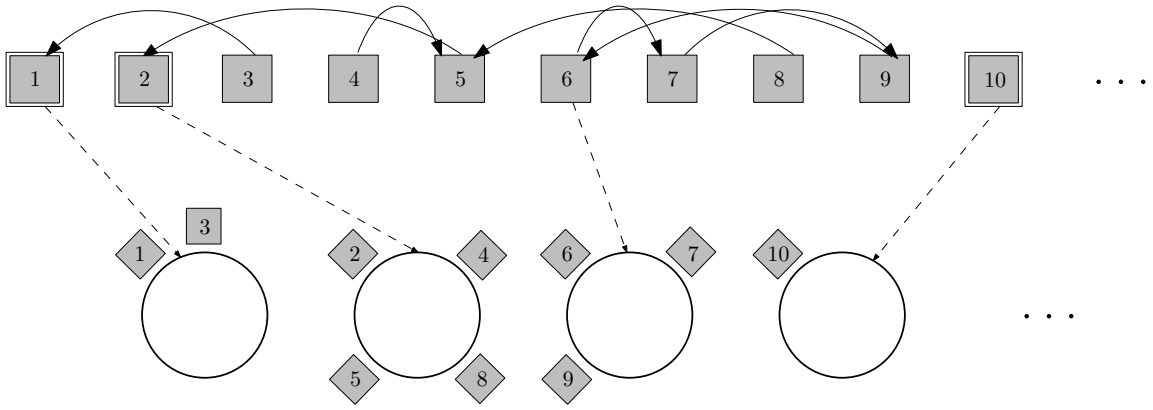

Figure 3.8: Illustration of the distance dependant Chinese restaurant process. Each customer chooses either one or none of the other costumers to sit with at the same table. Customers that have not chosen to sit with another person are indicated with a double box.

The distance dependant Chinese Restaurant Process

Whereas the traditional Chinese restaurant process allows costumers to sit around the tables in any ordering (no matter what permutation of their ordering, the probability of that certain setting is the same), in the *distance dependant Chinese restaurant processes* [Blei & Frazier, 2009], the random seating between customers depends on the distance between them. This distance can be spatial, temporal, or any other relevant characteristic. Therefore, instead of assigning customers to tables, the distance dependent CRP assigns customers to other customers. As a result, if two people are reachable by a sequence of costumers, they are seated at the same table. Figure 3.8 shows the customer to customer assignment for a sample distance dependent CRP along with the familiar table assignments for this sample.

The distance dependant CRP makes it possible to model many kinds of dependencies between data in infinite models. Blei & Frazier [2009] study how this dependency can be incorporated into language modelling applications.

The Indian Buffet Process

Another important extension of the CRP is the *Indian Buffet Process* (IBP) [Griffiths & Ghahramani, 2005; Ghahramani et al., 2007; Doshi, 2009]. In contrast to the CRP where the data points belong to one and only one cluster, in the Indian buffet process

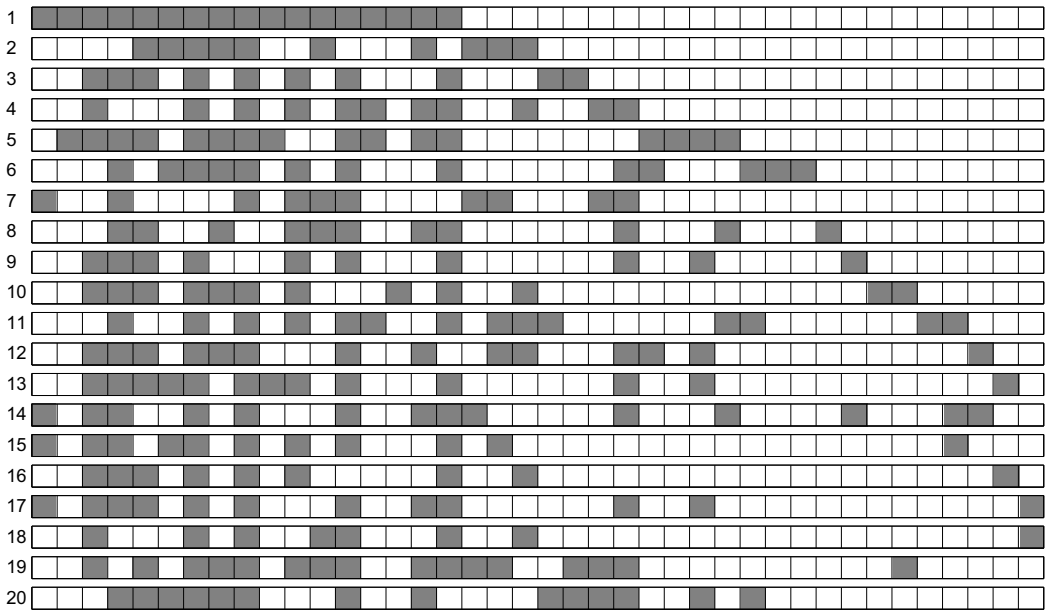

Figure 3.9: Samples from an Indian Buffet Process where the columns represent different features (possibly infinite) and the rows correspond to a finite number of observations. Figure from Griffiths & Ghahramani [2005]

each observation may belong to an infinite number of clusters at the same time. We can think of the clusters as latent features where each observation may be described by an infinite number of these latent variables. This can be regarded as a potentially infinite binary matrix as shown in Fig. 3.9.

The generative process for the Indian buffet process works as follows. Suppose there are n customers in a queue at an infinitely long Indian buffet. The first customer enters and tries the first Poisson(α)[1] dishes. For every costumer who later enters the restaurant, she samples from already sampled dishes with probability proportional to the number of people who have tried each dish, as well as Poisson($\frac{\alpha}{n}$) new dishes.

Although the IBP is considered as an infinite process, the construction shows that each customer almost surely has a finite number of dishes. The model described in this section can be used to infer an unknown number of latent features in data. For example, different symptoms of various diseases are usually modelled using an IBP, because the symptoms may be the same for various causes.

[1]The Poisson distribution is suitable for applications that involve counting the number of times a random event occurs in a given amount of time. Its probability mass function is given by $P\ (X\ =\ n)\ =\ \frac{e^{-\lambda}\ (\lambda)^n}{n!}$ where λ is the expected number of occurrences in the interval.

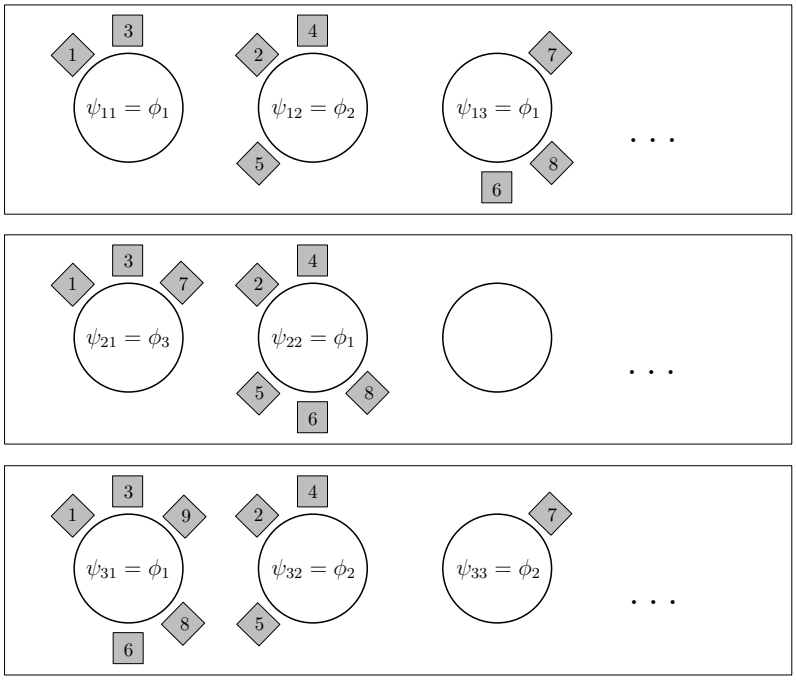

Figure 3.10: An illustration of the Chinese Restaurant Franchise, where the restaurants are depicted by a rectangle. Customers sit around the tables like the standard CRP, and at each table of each restaurant ψ_{ij} a dish is served from a global menu $\phi_1, \phi_2 \ldots$

The Chinese Restaurant Franchise

The *Chinese Restaurant Franchise* (CRF) [Teh et al., 2006] is one other extension of the CRP to multiple restaurants where they all share the same set of dishes. Like the Chinese restaurant process, customers enter the restaurant and chose their seats based on popularities similar to Eq. (3.16). However, the first costumer at each table of each restaurant orders a meal from an unbounded global menu $\phi_1, \phi_2 \ldots$ and this meal is shared among all the other customers sitting around that table, ψ_{ij}, where i is the restaurant number and j the specific table at that restaurant. Since the menu is global, multiple tables at different restaurants can order the same dish, as depicted in Fig. 3.10.

The construction of the CRF is such that the popular dishes get more popular. This has been proved to be useful in the detection and recognition of objects [Sudderth et al., 2006] as well as in Hierarchical Dirichlet Processes (HDP) [Teh et al., 2006] where the goal is to cluster grouped data using Dirichlet Process Mixture Models which we will explain in Section 3.3. In such settings, even though the data are separated into groups, but it is usually assumed that there are underlying links between data.

Group clustering problems occur frequently in practice, specially in the problem of topic discovery in document corpora.

3.2.5 Number of Unique Observations

In the previous sections, we discussed many important properties of the Dirichlet process and showed that instances from a DP are almost surely discrete (Theorem 3.2). We will conclude our discussion on the Dirichlet process by considering one illuminating aspect, specifically the number of unique observations K in a sequence of n samples. From Eq. (3.14) we know for the i'th sample, the probability of it having a new distinct value is equal to $\frac{\gamma}{\gamma + i - 1}$. By averaging over our n observations, we obtain:

$$E(K|n,\gamma) = \sum_{i=1}^{n} \frac{\gamma}{\gamma + i - 1} \approx \gamma \log\left(\frac{\gamma + n}{\gamma}\right) \tag{3.17}$$

where the approximation works when $n \to \infty$. In other words, the number of distinct observations grows only logarithmically in the number of samples. We have already seen this behaviour in Fig. 3.6, where even though we had 5000 samples, but number of distinct values turned out to be near 90 which is close to the number that the approximation in Eq. (3.17) returns.

Along with this important result, Antoniak [1974] also derived the distribution of the number of unique observations as:

$$p(K|n,\gamma) = \frac{\Gamma(\gamma)}{\Gamma(\gamma + n)} s(n,K) \gamma^K \tag{3.18}$$

Here $s(n,K)$ are the unsigned Stirling numbers of the first kind[1] and can be computed using recurrence formulae. However, the calculation of Stirling numbers using such formulae can be computationally inefficient when there are a large number of observations. For large n with $k = o(\log(n))$, Abramowitz & Stegun [1972] gave the following asymptotic approximation:

$$s(n,K) \approx \frac{(n-1)!}{(K-1)!} (\gamma_e + \log(n))^{K-1} \tag{3.19}$$

[1]The unsigned Stirling numbers of the first kind are the coefficients in the expression $x(x-1)\dots(x-n+1) = \sum_{k=1}^{n} s(n,k) x^k$. For a table of these numbers, as well as many of their interesting properties refer to Abramowitz & Stegun [1972].

where γ_e is the Euler - Mascheroni constant. Equation (3.18) is important for considering the implications of specific values of the concentration parameter γ on our prior over K.

In the next section, we will introduce an important statistical model based on the clustering effect of the CRP representation. This model is called the Dirichlet Process Mixture Model and is probably the most common application of the Dirichlet process [Teh, 2010].

3.3 The Dirichlet Process Mixture Model

In many situations, the observed data may be regarded as stemming from multiple populations. It is thus desirable to build models that can give us the ability to combine these samples. As an example, suppose we want to model the average number of words appearing in a document. If we consider the whole words of the document having the same underlying statistical model (one single cluster), our results and the inferences based on them may not be accurate. However, if we consider the document as comprising of different dialogue acts (one cluster for each dialogue act), and also each dialogue act to have its own statistical parameters, then we expect to get much better estimates for the quantities of interest. The statistical modelling of the Dihanna corpus, explained in Chapter 5, is based on this second approach.

The basic principle behind mixture models is to introduce and make inferences about a set of unobserved indicator variables $\mathbf{z} = (z_1, \ldots, z_n)$; given a set of observations $\mathbf{y} = (y_1, \ldots, y_n)$, the matrix of indicator variables $\mathbf{z}$ specifies the mixture component from which each of our observations is drawn from. In the following sections, we will first consider the case where we know the number of clusters of our mixture model a priori, and later, we incorporate our uncertainty in the true number of clusters by applying the clustering effect of the Chinese restaurant processes described in Section 3.2.4.

3.3.1 Finite Mixture Models

In Finite Mixture Models [McLachlan & Peel, 2000; Everitt & Hand, 1981; Gelman et al., 2004] we assume that our observations are drawn from K mixture components, but we don't know the component which underlies each observation. For every $1 \leq j \leq K$, we assume that the corresponding component has a density function f_j

which depends on the parameter vector $\boldsymbol{\theta_j}$. Furthermore, the parameter denoting the proportion of the population belonging to component i is denoted by λ_j with $\sum_{j=1}^{K} \lambda_j = 1$. In practice, we normally assume that the density function of the components belong to the same parametric family with only their parameter vectors being different from one another. Under such circumstances, the sampling distribution of a single observation y is:

$$p(y|\boldsymbol{\theta}, \boldsymbol{\lambda}) = \lambda_1 f(y|\boldsymbol{\theta_1}) + \lambda_2 f(y|\boldsymbol{\theta_2}) + \ldots + \lambda_K f(y|\boldsymbol{\theta_K}) \tag{3.20}$$

We can think of the mixing proportions $\boldsymbol{\lambda}$ as hyperparameters that determine the distribution of the indicator variables. Suppose we represent each z_i as a vector $\boldsymbol{\omega_i}$ such that:

$$\omega_{ij} = \begin{cases} 1 & \text{if } y_i \text{ is drawn from the j'th mixture component,} \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, each $\boldsymbol{\omega_i}$ has only one single non-zero element. We can think of the distribution over this latent variable vector to be Multinomial$(\lambda_1, \ldots, \lambda_m)$. Let $\boldsymbol{\Omega}$ be the $n \times K$ matrix composed of the vectors $\boldsymbol{\omega_i}$, it then follows that the joint distribution of the data and the indicator variables is:

$$p(\mathbf{y}, \boldsymbol{\Omega}|\boldsymbol{\Theta}, \boldsymbol{\lambda}) = p(\boldsymbol{\Omega}|\boldsymbol{\lambda}) p(\mathbf{y}|\boldsymbol{\Omega}, \boldsymbol{\Theta}) = \prod_{i=1}^{n} \prod_{j=1}^{K} (\lambda_j f(y_i|\boldsymbol{\theta_j}))^{\omega_{ij}} \tag{3.21}$$

where $\boldsymbol{\Theta}$ is the matrix of parameter vectors; i.e. each row contains the parameters of the corresponding mixture component distribution.

Prior Distribution for the Parameters

There are two parameters in a mixture model as described above; the mixing proportions $\boldsymbol{\lambda} = (\lambda_1, \ldots \lambda_K)$ and the matrix of parameters of the probability distribution function corresponding to each component $\boldsymbol{\Theta}$. As we described earlier, the indicator variable vector $(\omega_{i1}, \ldots, \omega_{iK})$ of each observation is thought of as a random variable with distribution Multinomial$(\lambda_1, \ldots, \lambda_K)$. Therefore, a natural choice for the prior on $\boldsymbol{\lambda}$ is the conjugate prior distribution of the multinomial distribution; the Dirichlet distribution $\boldsymbol{\lambda} \sim$ Dirichlet$(\alpha_1, \ldots, \alpha_K)$. As we know from the discussion on the Dirichlet distribution in Section 2.3.3, the α_j determine the mean of the prior distribution whereas their sum $s = \sum_i (\alpha_i)$ is a measure of the concentration or strength of the prior.

In general, there are no restrictions on the prior for the component parameter matrix $\boldsymbol{\Theta}$. However, a popular choice is the Dirichlet distribution as we will describe below using an over simplified example. Before that, we will summarize these discussions and give a hierarchical representation for the finite mixture model which will help better understand the structure and interdependence of model parameters.

Hierarchical Representation

With the discussions in this section, we can give the following hierarchical representation for a finite mixture model:

$$\begin{aligned} y_i|\mathbf{z}, \boldsymbol{\Theta} &\sim f(y_i|\boldsymbol{\theta}_{z_i}) && 1 \leq i \leq n \\ z_i|\boldsymbol{\lambda} &\sim \text{Multinomial}(\lambda_1, \ldots, \lambda_K) \\ \boldsymbol{\theta_j} &\sim G_0 && 1 \leq j \leq K \\ \boldsymbol{\lambda} &\sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_K) \end{aligned} \tag{3.22}$$

where $\boldsymbol{\theta}_{z_i}$ refers to the parameter vector corresponding to mixture component z_i. We can equivalently rewrite the finite mixture model in terms of draws from a discrete distribution G^K and then sampling the parameter matrix from this distribution as represented below:

$$\begin{aligned} y_i|\hat{\boldsymbol{\theta}} &\sim f(y_i|\hat{\boldsymbol{\theta}}_{\boldsymbol{i}}) \\ \hat{\boldsymbol{\theta}}_{\boldsymbol{i}} &\sim G^K \\ G^K(\boldsymbol{\theta}) &= \sum_{j=1}^{K} \lambda_j \delta_{\boldsymbol{\theta_j}} \\ \boldsymbol{\lambda} &\sim \text{Dirichlet}(\alpha/K, \ldots, \alpha/K) \\ \boldsymbol{\theta_j} &\sim H_0 \qquad 1 \leq j \leq K \end{aligned} \tag{3.23}$$

It can be shown that these two representations are statistically equivalent (see e.g. Ishwaran & Zarepour [2002] or Sudderth [2006]). Later, we will show that the representation in Eq. (3.23) provides a way to approximate the Dirichlet process mixture model that we will explain in the next section.

To help clarify the generative process of building a finite mixture model based on Eqs. (3.22), consider the following illustrative example. Suppose we have a dialogue which is comprised of only two types of utterances; U1 with parameter θ_1 and U2 with parameter θ_2. Also, suppose that our vocabulary dictionary simply contains the words A, B and C. To construct the model, we first draw the mixing proportions from a Dirichlet distribution with parameters estimated as 0.5 and 1; i.e. $\lambda_1,\ \lambda_2 \sim \text{Dirichlet}(1, 3)$. What this means

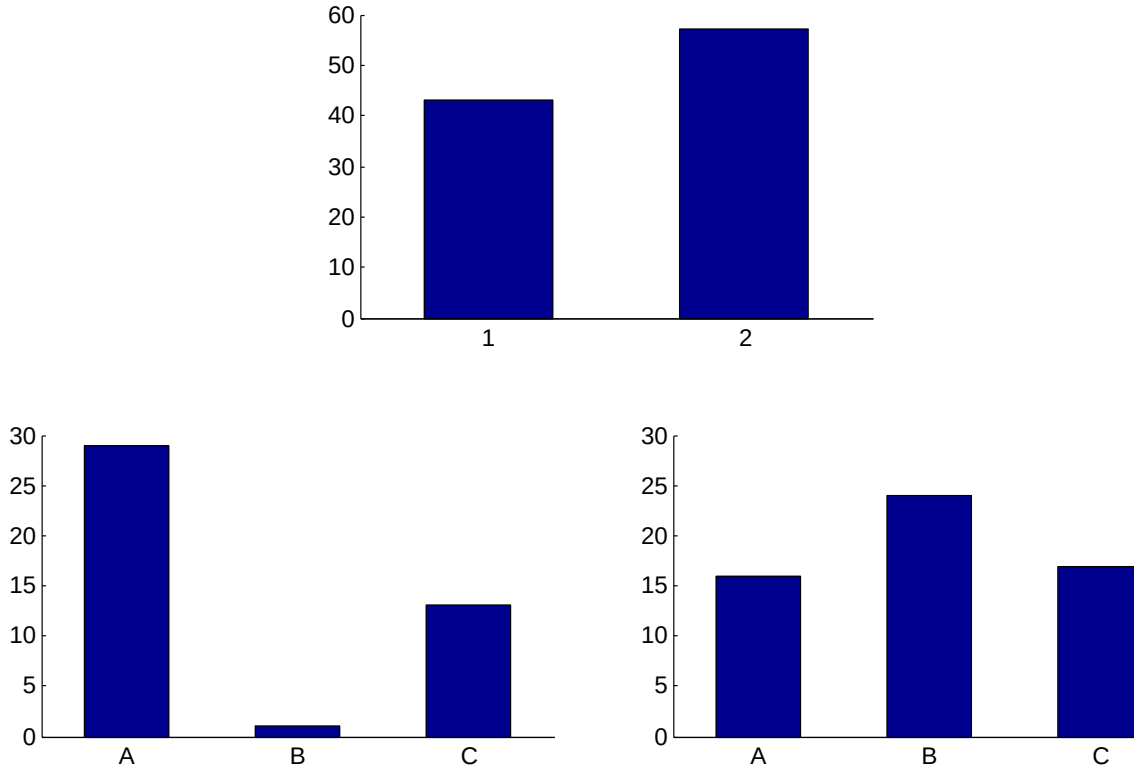

Figure 3.11: An example of a finite mixture model. The top plot depicts 100 multinomial samples from a prior over components with distribution Dirichlet$(1, 3)$. The two bottom plots illustrate sample words for each of the drawn sample when the parameters of the first and second component are drawn from $\theta_1 \sim$ Dirichlet$(3, 1, 1)$ and $\theta_2 \sim$ Dirichlet$(1, 3, 1)$, respectively.

is that most of our samples belong to U2 because the mean of the Dirichlet distribution is closer to the second utterance, as depicted in the top plot of Fig. 3.11.

The next step is to draw the parameters of each utterance, θ_1 and θ_2 from an appropriate distribution. As we will fully explain in Chapter 5, by modelling utterances using the 'bags of words' model, it is rational to regard each utterance as a Multinomial distribution over the vocabulary space, and thus a good choice of G_0 would be its conjugate prior distribution, the Dirichlet distribution. Again using some estimation techniques, suppose we figure out that the parameter vector of the components are drawn from $\theta_1 \sim$ Dirichlet$(3, 1, 1)$ and $\theta_2 \sim$ Dirichlet$(1, 3, 1)$, respectively. Therefore by assuming that we have 5 observations, a hierarchical representation for the model described here, similar to that of Eq. (3.22), would be as follows:

$$\begin{aligned} y_i &\sim \text{Multinomial}(\boldsymbol{\theta}_{z_i}) \qquad 1 \leq i \leq 5 \\ z_i &\sim \text{Multinomial}(\lambda_1, \lambda_2) \\ \theta_1 &\sim \text{Dirichlet}(3, 1, 1) \\ \theta_2 &\sim \text{Dirichlet}(1, 3, 1) \\ \lambda_1, \lambda_2 &\sim \text{Dirichlet}(1, 3) \end{aligned} \tag{3.24}$$

Fig. 3.11 shows 100 multinomial samples from this model. As we discussed above, utterances that belong to U1 will have more A's than B or C, whereas utterances from U2 will have more B's than A or C.

Following the steps described above, we first draw a component $z_i \sim \text{Multinomial}(\lambda_1, \lambda_2)$ for each utterance y_i, and then based on the component that it belongs to, the words in each utterance are generated from repeated draws from the corresponding component distribution $\text{Multinomial}(\theta_{z_i})$. For example, we may have the following draws $y_1 = AABCA$, $y_2 = BABCB$, $y_3 = BBBA$, $y_4 = BAABCA$ and $y_5 = ACBBB$ where y_1 and y_4 belong to the first component while y_2, y_3 and y_5 are from the second component.

3.3.2 Infinite Mixture Models

We can make use of the nonparametric nature of the Dirichlet process to generate mixture models with an infinite number of clusters. As with finite mixture models, assume that we have a hierarchical model in which each of our observations y_i has a density function f with independently sampled parameter vector $\boldsymbol{\theta_i}$. However, in the limit $K \to \infty$, the corresponding labels lose their meaning as the space of possible labels becomes continuous [Ranganathan, 2006]. This means that we can discard the labels and since the infinite limit of a Dirichlet distribution is a Dirichlet process, the infinite dimensional counterpart of the mixture model explained above would be as follows:

$$\begin{aligned} y_i|\boldsymbol{\Theta} &\sim f(y_i|\boldsymbol{\theta_i}) \\ \boldsymbol{\theta_i}|G &\sim G \\ G|\gamma, H &\sim \text{DP}(\gamma, H) \end{aligned} \tag{3.25}$$

This is called the Dirichlet Process Mixture Model (DPMM) [Antoniak, 1974; Ranganathan, 2006; Neal, 2000; Sudderth, 2006]. To see that it is actually the infinite limit of the finite mixture model, suppose we place a symmetric Dirichlet prior over the mixing proportions in Eq. (3.22):

$$\lambda_1, \ldots, \lambda_K \sim \text{Dirichlet}(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K}) \tag{3.26}$$

Now consider the marginal probability of the indicator variables given the previous values by applying Eq. (2.12):

$$p(z_i = c_j | z_1, \ldots, z_{i-1}) = \int_\Theta p(z_i = c_j | \boldsymbol{\lambda}) p(\boldsymbol{\lambda} | z_1, \ldots, z_{i-1}) \tag{3.27}$$

where c_j refers to the j'th component, thus it follows that $p(z_i = c_j | \boldsymbol{\lambda}) = \lambda_j$. Furthermore, we know that $\boldsymbol{\lambda}$ follows a Dirichlet distribution, and by the Dirichlet-Multinomial conjugacy, the posterior is also a Dirichlet with the vector of sufficient statistics updated as in Eq. (2.35). Therefore, we can simplify Eq. (3.27) as follows:

$$\begin{aligned} p(z_i = c_j | z_1, \ldots, z_{i-1}) &= \int_\Theta \lambda_j p(\boldsymbol{\lambda} | z_1, \ldots, z_{i-1}) && (3.28) \\ &= E(\lambda_j | z_1, \ldots, z_{i-1}) && (3.29) \\ &= \frac{n_j + \alpha/K}{\alpha + i - 1} && (3.30) \end{aligned}$$

where the last line follows from Eq. (2.32).

Now if we take the limit $K \to \infty$, the conditional probability in Eq. (3.30) reach the following limits:

$$\lim_{K \to \infty} p(\boldsymbol{z_i} = c_j | z_1, \ldots, z_{i-1}) = \begin{cases} \dfrac{n_j}{\alpha + i - 1} & \text{if } 1 \leq j \leq K \\ \dfrac{\alpha}{\alpha + i - 1} & \text{if } j = K + 1 \end{cases} \tag{3.31}$$

Comparing this result with Eq. (3.16) shows that in the limit $m \to \infty$, the finite mixture model explained above is the same as the Dirichlet process mixture model described in Eq. (3.25). The following theorem from Ishwaran & Zarepour [2002] makes use of this result and suggests that in this limit, predictions based on the finite mixture model approach those of the corresponding Dirichlet process mixture model.

Theorem 3.4. *Let H denote a probability measure on $\boldsymbol{\Theta}$ and f a real valued measurable function with domain $\boldsymbol{\Theta}$. Consider the discrete distribution G^K and it's corresponding mixture model as described in Eq. (3.23). As $K \to \infty$, expectations with respect to G^K converge in distribution to a corresponding Dirichlet process:*

$$\int_{\boldsymbol{\Theta}} f(\theta) dG^K(\theta) \to \int_{\boldsymbol{\Theta}} f(\theta) dG(\theta) \qquad G \sim DP(\gamma, H) \tag{3.32}$$

We can apply Theorem 3.4 to approximate a DP mixture model by finite means. Ishwaran & Zarepour [2000b] suggested a Gibbs sampling algorithm for approximate learning of the model, however, they noted that their approximation converges slowly with

the number of clusters K and a large number of potential clusters may be required [Ishwaran & Zarepour, 2002; Sudderth, 2006]. More accurate stochastic approximations based on the *truncated stick-breaking representation* have been proposed by Ishwaran & James [2001] and Ishwaran & Zarepour [2000b], as well as deterministic variational alternatives by Blei & Jordan [2006] and Kurihara et al. [2007].

3.3.3 Sampling from the Dirichlet Process Mixture

Given a set of observations $\mathbf{y} = (y_1, \ldots, y_n)$ and the parameters γ and H, the Dirichlet process mixture model of Eq. (3.25) gives a posterior distribution for the matrix of indicator variables $\boldsymbol{\Theta}$. However, the discreteness of the random measure G makes exact calculation of this posterior mathematically intractable due to the need to take into account all possible configurations that the $\boldsymbol{\theta_i}$'s are identical and distinct (see Antoniak [1974] for a thorough discussion). Therefore, we have to resort to approximation methods to calculate the posterior.

Among different approximation methods, MCMC techniques and in particular, the Gibbs sampler, have gained widespread popularity. Below we will describe a general framework for the Gibbs sampler which we have also used for our experiments in Chapter 5.

Consider the predictive distribution for the indicator variables as mentioned in Eq. (3.16). Since the joint distribution of the indicators does not depend on their ordering, for each z_i we condition not only on the previous values $z_1, \ldots, z_{i-1}$, but on all z_j from 1 to n except z_i itself. Let $z_{(-i)}$ stand for this vector. Our goal is to find the posterior on z_i given $z_{(-i)}$ and a data instance y_i, as well as the Dirichlet process parameters. To do this, consider that for any set of events A, B and C where B and C are independent, we have:

$$p(A|B,C) = \frac{p(C|A,B)p(A|B)}{p(C)} \tag{3.33}$$

This simple probability fact can be easily obtained from Bayes' theorem (Theorem 2.1) and the definition of conditional probability in Eq. (2.6). Now by applying Eq. (3.33), we can write the conditional distribution of our posterior in unscaled form as:

$$p(z_i|z_{(-i)}, y_i, \gamma, H) \propto p(y_i|\mathbf{z})p(z_i|z_{(-i)}, \gamma, H) \tag{3.34}$$

From Eq. (3.25), we know that the likelihood $p(y_i|\mathbf{z}) = f(y_i|\mathbf{z})$. As for the second term,

from Eq. (3.16) we have:

$$p(z_i|z_{(-i)}, \gamma, H) = \frac{1}{\gamma + n - 1}\left(\gamma H + \sum_{j \neq i} \delta_{z_j}\right) \tag{3.35}$$

Now by combining these equations, we get the following conditional distribution for use in Gibbs sampling:

$$p(z_i|z_{(-i)}, y_i, \gamma, H) = b\gamma H(z_i) f(y_i|z_i) + b\sum_{j \neq i} f(y_i|z_j)\delta_{z_j} \tag{3.36}$$

where b is the normalizing constant and is equal to:

$$b = \left(\gamma \int_{\boldsymbol{z}} H(z) f(y_i|z) dz + \sum_{j \neq i} f(y_i|z_j)\right)^{-1} \tag{3.37}$$

It can be observed that the term $\int_{\boldsymbol{z}} H(z) f(y_i|z) dz$ in Eq. (3.36) is the marginal distribution of y_i, thus we can rewrite the equation in terms of the posterior f_p

$$p(z_i|z_{(-i)}, y_i, \gamma, H) = \left(b\gamma \int_{\boldsymbol{z}} H(z) f(y_i|z) dz\right) f_p(z_i|y_i) + b\sum_{j \neq i} f(y_i|z_j)\delta_{z_j} \tag{3.38}$$

where f_p is equal to:

$$f_p(z_i|y_i) = \frac{H(z) f(y_i|z) dz}{\int_{\boldsymbol{z}} H(z) f(y_i|z) dz} \tag{3.39}$$

Based on these equations, Escobar [1994] proposed the simple Gibbs sampler described in Algorithm 2 (see also [Escobar & West, 1995]).

For the sampling algorithm to be feasible, calculation of the integral and sampling from f_p must be computationally tractable. This is normally achieved when H is chosen to be the conjugate prior of f_p. Based on this clever sampling algorithm, MacEachern [1994] developed an alternative sampling strategy which most of the later sampling algorithms are based on this idea. A more detailed description of this algorithm can be found in MacEachern & Muller [1998]. Further work and discussion on a variety of sampling algorithms, even for the case where non-conjugate priors are used, can be found in [Neal, 2000].

Algorithm 2: Gibbs sampling from $p(z_i|z^j_{(-i)}, y_i, \gamma, H)$

Choose a starting point $\mathbf{z}^0 = (\mathbf{z}^0_1, \mathbf{z}^0_2, \ldots, \mathbf{z}^0_n)$;
Generate new samples as follows
for $j + 1 = 1$ **to** T **do**
  for $i = 1$ **to** n **do**
    Sample z_i^{j+1} from the posterior distribution $p(z_i|z^j_{(-i)}, y_i, \gamma, H)$ as follows:
    begin
      $z_i^{j+1} = z_l^j$, $l \neq i$, with probability $bf(y_i|z_l^j)$
      $z_i^{j+1} \sim f_p(z|y_i)$ with probability $b\gamma \int_{\boldsymbol{z}} H(z)f(y_i|z)dz$
    end
  end
end

3.4 Summary

In this chapter, we discussed the principles of nonparametric Bayesian methods. Among all nonparametric approaches, we gave a detailed overview of the Dirichlet process with its properties demonstrated through its different representations. Among these processes, the Chinese restaurant process is the most widely used representation due to its emphasis on the clustering effect of the DP. Many variations of the CRP have been proposed in the literature, among which we discussed the nested CRP, the distance dependent CRP, the Indian buffet and the Chinese Restaurant Franchise. We also covered finite mixture models and their infinite dimensional counterpart, the Dirichlet process mixture model.

The last section was devoted to a Gibbs sampler for the DPMM which we will use for the experiments in Chapter 5. The next chapter will elaborate on both finite and infinite mixture models and provide means for estimating hyperparameters in these models.

Chapter 4

Hyperparameter Estimation for Mixture Models

In classification applications using mixture models that deal with text data, such as the problem of dialogue act classification, it is normally assumed that the appearance of words in each component follows a multinomial distribution with parameters governed by a Dirichlet distribution prior (or a Dirichlet process, in the case of infinite mixture models). However, this choice of prior is strongly dependent on the hyperparameters of the corresponding finite or infinite dimensional Dirichlet distribution. In this chapter, we will discuss some of the most important methods devised for this purpose, along with two new improvements for constraint-based estimation of the hyperparameters in the finite dimensional case.

4.1 Finite Mixture Hyperparameter Estimation

In Section 3.3.1 we introduced the finite mixture model and gave an example in the domain of text modelling to demonstrate how these models work. There we mentioned that it is usual to assume a Multinomial distribution for the words appearing in each utterance, and a natural choice for the prior would be its conjugate distribution, namely the Dirichlet distribution. This combination is often called the Dirichlet-Multinomial distribution which we will start this section by giving a brief overview of its most important properties.

4.1.1 The Dirichlet-Multinomial Distribution

The Dirichlet-Multinomial distribution, also called the Multivariate Polya Distribtion (MPD) [Kvam & Day, 2001; Minka, 2003] or the Dirichlet Compound Multinomial (DCM) [Madsen et al., 2005; Elkan, 2006], is a hierarchical model where the observations are drawn from a multinomial distribution whose parameters are generated by a Dirichlet distribution. They normally arise in applications where we have to estimate probabilities from count data; these include topic modelling [Steyvers & Griffiths, 2007], military combat modelling where targets are randomly assigned to platforms [Kvam & Day, 2001], or biological sequence analysis where different nucleotides occur at different positions in a DNA sequence [Durbin, 1998].

Suppose we have a set of observations $D = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ each composed of a vector of counts $\mathbf{c}_i$, $i = 1, \ldots, n$. Following the notation in Eq. (3.22) we can represent this model hierarchically as follows:

$$\begin{aligned} \mathbf{y}_i = (c_{i1}, \ldots, c_{iK})^T | \boldsymbol{\theta} &\sim \text{Multinomial}(\theta_1, \ldots, \theta_K) \\ \boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)^T | \boldsymbol{\beta} &\sim \text{Dirichlet}(\beta_1, \ldots, \beta_K) \end{aligned} \tag{4.1}$$

The main intuition for using the Dirichlet-Multinomial distribution in text and dialogue modelling is that each document (or utterance) can be thought of as a multinomial distribution over the set of possible words, which this multinomial is itself governed by a Dirichlet prior. Here the Dirichlet represents a general topic, while the multinomial makes the words likely for the document chosen. For example, suppose we have a news feed about soccer events in Europe, some generated from a multinomial distribution which gives high probability to Italian soccer team names, others emphasizing on the English Premier League teams. By using a Dirichlet prior, this prior distribution which represents the whole news feed, gives high probability to the multinomials representing these two leagues, while giving low probability to other leagues such as the German Bundes Liga.

Let us denote the sum of counts for the observation vector i by n_i, and the total number of counts in component k by m_k. In other words:

$$n_i = \sum_{k=1}^{K} c_{ik} \tag{4.2}$$

$$m_k = \sum_{i=1}^{n} c_{ik} \tag{4.3}$$

Therefore, from Eq. (3.30) we can write the predictive probability of observing outcome k in the context of the d'th experiment as follows:

$$p(k|d, D, \boldsymbol{\beta}) = \frac{c_{dk} + \beta_k}{n_d + \sum_{i=1}^{K} \beta_i} \tag{4.4}$$

We will now derive some important properties of the Dirichlet-Multinomial distribution which will be used in the rest of the thesis.

Marginal Distribution

In most applications, even though we talk about the multinomial parameters $\boldsymbol{\theta}$, the only actual parameters that are important in our modelling are the parameters of the Dirichlet prior $\boldsymbol{\beta}$. Therefore it is desirable to calculate the probability of our observations with the multinomial parameters marginalized out. According to Eq. (2.8) this probability can be written as follows:

$$p(\mathbf{y}_i|\boldsymbol{\beta}) = \int_{\Theta} p(\mathbf{y}_i|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\boldsymbol{\beta}) d\boldsymbol{\theta} \tag{4.5}$$

According to our hierarchical model as expressed in Eq. (4.1), we assume that $p(\mathbf{y}|\boldsymbol{\theta})$ follows a multinomial distribution and $p(\boldsymbol{\theta}|\boldsymbol{\beta})$ follows a Dirichlet distribution. We can thus simplify Eq. (4.5) by replacing their corresponding probability function:

$$p(\mathbf{y}_i|\boldsymbol{\beta}) = \int_{\Theta} \left(\frac{n_i!}{\prod_{k=1}^{K} c_{ik}!} \prod_{k=1}^{K} \theta_k^{c_{ik}} \right) \left(\frac{\Gamma(\sum_{k=1}^{K} \beta_k)}{\prod_{k=1}^{K} \Gamma(\beta_k)} \prod_{k=1}^{K} \theta_k^{\beta_k - 1} \right) d\boldsymbol{\theta} \tag{4.6}$$

$$= \frac{n_i!}{\prod_{k=1}^{K} c_{ik}!} \frac{\Gamma(\sum_{k=1}^{K} \beta_k)}{\prod_{k=1}^{K} \Gamma(\beta_k)} \int_{\Theta} \prod_{k=1}^{K} \theta_k^{c_{ik} + \beta_k - 1} d\boldsymbol{\theta} \tag{4.7}$$

It can be observed that the term inside the integral of Eq. (4.7) is the unnormalized version of the Dirichlet distribution with parameters $(\mathbf{y}_i + \boldsymbol{\beta})$, and using the fact that integrating a distribution over all possible parameter values must yield 1, the integral of Eq. (4.7) has to be equal to the inverse of the normalizing factor. In other words:

$$\int_{\Theta} \text{Dirichlet}(\mathbf{y}_i + \boldsymbol{\beta}) = \frac{\Gamma\left(\sum_{k=1}^{K} c_{ik} + \beta_k\right)}{\prod_{k=1}^{K} \Gamma(c_{ik} + \beta_k)} \int_{\Theta} \prod_{k=1}^{K} \theta_k^{c_{ik} + \beta_k - 1} d\boldsymbol{\theta} = 1 \tag{4.8}$$

So if we replace the integral in Eq. (4.7) with the inverse of the coefficient of the same integral in Eq. (4.8), we can easily get:

$$p(\mathbf{y}_i|\boldsymbol{\beta}) = \frac{\Gamma(n_i+1)\Gamma\left(\sum_{k=1}^{K}\beta_k\right)}{\Gamma\left(\sum_{k=1}^{K}c_{ik}+\beta_k\right)}\prod_{k=1}^{K}\frac{\Gamma(c_{ik}+\beta_k)}{\Gamma(c_{ik}+1)\Gamma(\beta_k)} \tag{4.9}$$

where we have replaced the factorials with Gamma functions using the fact that $\Gamma(n+1) = n!$ for positive integers, thus transforming Eq. (4.9) into products of only Gamma functions. In order to simplify calculations, it is normally preferable to work with the logarithm of the marginal distribution which can be written as:

$$\begin{aligned}\log p(\mathbf{y}_i|\boldsymbol{\beta}) = & \\ & \log\Gamma(n_i+1) + \log\Gamma\left(\sum_{k=1}^{K}\beta_k\right) - \log\Gamma\left(\sum_{k=1}^{K}(c_{ik}+\beta_k)\right) \\ & + \sum_{k=1}^{K}\left(\log\Gamma(c_{ik}+\beta_k) - \log\Gamma(c_{ik}+1) - \log\Gamma(\beta_k)\right)\end{aligned} \tag{4.10}$$

Using the logarithm probability has two main advantages; first, the logarithm of the Gamma function grows much slower and converts computationally demanding multiplication operations into simple additions and subtractions. Also, in many programming languages, there are functions which approximate the logarithm of the Gamma function, mainly based on formulae proposed by Rocktaeschel [1922] and Bohmer [1939] (in MATLAB, this function is called the gammaln function). Second, for the maximum likelihood estimate of the hyperparameters that we will discuss in the remainder of this chapter, finding the derivative of the logarithm is much easier than working with the original function as expressed in Eq. (4.9). We will get back to this issue later in Section 4.1.2.

Predictive Distribution

Suppose we are interested in drawing conclusions about potential values of a new observation $\mathbf{y}_f = (c_{f1}, \ldots, c_{fK})^T$ which is independent of the previous observations $D = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$. This distribution, which is the probability of a future observation conditional on previous observations, should be calculated using Eq. (2.12) by evaluating the integral:

$$p(\mathbf{y}_f|\mathbf{y}_1, \ldots, \mathbf{y}_n, \boldsymbol{\beta}) = \int_{\Theta} p(\mathbf{y}_f|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{y}_1, \ldots, \mathbf{y}_n, \boldsymbol{\beta})d\boldsymbol{\theta} \tag{4.11}$$

By comparing the right hand sides of Eqs. (4.11) and (4.4), it can seen that the only difference between the two integrands is that in the case of prediction, we must deal with the posterior Dirichlet distribution instead of the original Dirichlet prior. In other words, instead of drawing $\boldsymbol{\theta} \sim \text{Dirichlet}(\boldsymbol{\beta})$, here we have to draw $\boldsymbol{\theta} \sim \text{Dirichlet}(\boldsymbol{\beta} + \mathbf{m})$ where the vector $\mathbf{m}$ is defined as:

$$\mathbf{m} = (m_1, m_2, \ldots, m_K)^T \tag{4.12}$$

and the m_k are the same as in Eq. (4.2). With the same argument, it thus follows that the predictive distribution is equal to:

$$p(\mathbf{y}_f|\mathbf{y}_1, \ldots, \mathbf{y}_n, \boldsymbol{\beta}) = \frac{\Gamma(n_f + 1)\Gamma\left(\sum_{k=1}^{K} m_k + \beta_k\right)}{\Gamma\left(\sum_{k=1}^{K} c_{fk} + m_k + \beta_k\right)} \prod_{k=1}^{K} \frac{\Gamma(c_{fk} + m_k + \beta_k)}{\Gamma(c_{fk} + 1)\Gamma(m_k + \beta_k)} \tag{4.13}$$

For the same reasons discussed previously, it is usually more convenient to work with the logarithm of the predictive distribution which can be written as follows:

$$\begin{aligned}\log p(\mathbf{y}_f|\mathbf{y}_1, \ldots, \mathbf{y}_n, \boldsymbol{\beta}) = \\ \log\Gamma(n_f + 1) + \log\Gamma\left(\sum_{k=1}^{K}(m_k + \beta_k)\right) - \log\Gamma\left(\sum_{k=1}^{K}(c_{fk} + m_k + \beta_k)\right) \\ + \sum_{k=1}^{K}\left(\log\Gamma(c_{fk} + m_k + \beta_k) - \log\Gamma(c_{fk} + 1) - \log\Gamma(m_k + \beta_k)\right)\end{aligned} \tag{4.14}$$

Therefore, from the above discussions, the only hyperparameters that need to be estimated for making any sort of Bayesian inference in finite mixture models are the parameters of the Dirichlet prior. In the rest of this chapter, we will cover some of the basic estimation techniques for obtaining estimates in these models.

4.1.2 Hyperparameter Estimation Techniques

Suppose we have a set of i.i.d observations $D = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ drawn from a Dirichlet-Multinomial distribution as described in the previous section. In this case, the log likelihood of the dataset given the Dirichlet hyperparameters, $L(\boldsymbol{\beta})$, can be written as:

$$L(\boldsymbol{\beta}) = \log p(D|\boldsymbol{\beta}) = \sum_{i=1}^{n} \log p(\mathbf{y}_i|\boldsymbol{\beta}) \tag{4.15}$$

where $\log p(\mathbf{y}_i|\boldsymbol{\beta})$ is given in Eq. (4.10). To find the maximum likelihood estimate of the hyperparameters, we normally try to find the roots to the derivative of the likelihood function; i.e. to solve the following equation:

$$\nabla L(\boldsymbol{\beta}) = (g_1(\boldsymbol{\beta}), \ldots, g_K(\boldsymbol{\beta}))^T = 0 \tag{4.16}$$

where each component $g_k(\boldsymbol{\beta})$ is given by:

$$g_k(\boldsymbol{\beta}) = \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} = \sum_{i=1}^{n} \left(\psi(c_{ik} + \beta_k) - \psi(\beta_k) + \psi\left(\sum_{k=1}^{K} \beta_k\right) - \psi\left(\sum_{k=1}^{K} (c_{ik} + \beta_k)\right)\right) \tag{4.17}$$

and ψ is the digamma function[1]. Since we will be dealing with Eq. (4.17) many times in the following sections, it is better to simplify the equation and write it as:

$$\frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} = L_1(\beta_k) + L_2(\boldsymbol{\beta}) \tag{4.18}$$

where L_1 is the part of g_k which depends only on the parameter β_k, and L_2 depends on β_k only through the sum of the vector $\boldsymbol{\beta}$:

$$\begin{aligned} L_1(\beta_k) &= \sum_{i=1}^{n} (\psi(c_{ik} + \beta_k) - \psi(\beta_k)) \\ L_2(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \left(\psi\left(\sum_{k=1}^{K} \beta_k\right) - \psi\left(\sum_{k=1}^{K} (c_{ik} + \beta_k)\right)\right) \end{aligned} \tag{4.19}$$

Unfortunately, there is no closed form solution for the zeros of Eq. (4.17), thus we have to resort to numerical optimization techniques.

Most of the literature on finding the optimal hyperparameter values of a Dirichlet-Multinomial distribution are based on the works of Minka [2003]. However, as noted by Huang [2005], the proposed methods lack an important nontrivial implementation constraint; that the hyperparameters of the Dirichlet prior must be positive. This is true for all Minka's algorithms except for his fixed point iteration method which itself enforces this condition due to its construction. Below we will improve two of Minka's algorithms by applying constraint based optimization techniques. To the writers knowledge

[1] The digamma function is the logarithmic derivative of the Gamma function, in other words: $\psi(x) = \frac{d}{dx} \log \Gamma(x)$.

no previous attempt has been made to incorporate constraints on the hyperparameters into estimation techniques. We begin with exponential mapping for Newton–Raphson iterations, and formulae for finding the optimal estimators.

Newton–Raphson using an Exponential Mapping

The Newton–Raphson method is a widely used, powerful method for finding the roots of a function. Here we will give a very brief overview of this method, interested readers may refer to Burden & Faires [2010], Kreyszig [2007] or Yang [2008] for complete explanation along with numerous examples.

The multivariate version of the Newton–Raphson method attempts to find the maxima or minima of functions of multiple variables:

$$f(\mathbf{x}) = f(x_1, \ldots, x_K) \tag{4.20}$$

The extrema of f can be obtained by finding the solutions to the equation $\nabla f(\mathbf{x}) = 0$. It can be shown that under certain easily-met conditions (see e.g. Burden & Faires [2010]), the following iteration scheme will converge to an extremum point of $f(\mathbf{x})$:

$$\mathbf{x}^{\text{new}} = \mathbf{x}^{\text{old}} - \left[\mathbf{H}f(\mathbf{x}^{\text{old}})\right]^{-1} \nabla f(\mathbf{x}^{\text{old}}) \tag{4.21}$$

where $\mathbf{H}f(\mathbf{x})$ is the Hessian matrix of $f(\mathbf{x})$ defined as:

$$\mathbf{H}f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_K} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_K} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_K \partial x_1} & \frac{\partial^2 f}{\partial x_K \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_K^2} \end{bmatrix} \tag{4.22}$$

Now let us return to the original problem in which we were trying to maximize the MLE by finding the roots of Eq. (4.16). As we noted in Section 2.3.3, here we are constrained to $\boldsymbol{\beta} > 0$ because the parameters of the Dirichlet distribution must always remain positive. This constraint must be satisfied not only in the final solution, but also in every step of our iteration scheme.

There are two ways to enforce this condition. In this section, we will approach this goal by mapping each component of our parameter vector into an exponential component, in other words by using the following mapping:

$$(\beta_1, \ldots, \beta_K)^T \longmapsto (e^{\tau_1}, \ldots, e^{\tau_K})^T \tag{4.23}$$

The advantage of this mapping is that no matter what the choice of the parameters τ_k, we will always have a positive value for β_k. It is obvious that we can get from one representation to the other by:

$$\beta_k = e^{\tau_k}, \qquad \text{and} \qquad \tau_k = \log \beta_k \tag{4.24}$$

Therefore, our goal is now to find the optimal values of $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_K)^T$ that maximize Eq. (4.15) with this new parametrization; i.e. maximize $I(\boldsymbol{\tau}) = L(e^{\tau_1}, \ldots, e^{\tau_K})$. This implies that we should find the gradient and inverse Hessian matrix of $I(\boldsymbol{\tau})$. To do this, we use a simple trick inspired by the equalities in Eq. (4.24) and instead of differentiating the latter function with respect to $\boldsymbol{\tau}$, we differentiate the original function $L(\boldsymbol{\beta})$ with respect to $\log \boldsymbol{\beta}$ and reparametrize after having finished the calculations. For the gradient, this amounts to the following equation for each of the components:

$$\frac{\partial I(\boldsymbol{\tau})}{\partial \tau_k} \overset{[\tau \mapsto \log \beta]}{\equiv} \frac{\partial L(\boldsymbol{\beta})}{\partial \log \beta_k} = \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} \left(\frac{\partial \log \beta_k}{\partial \beta_k} \right)^{-1} = \beta_k \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} \tag{4.25}$$

where the symbol $\overset{[\tau \mapsto \log \beta]}{\equiv}$ means that the two sides are equivalent under the mapping $\tau \longmapsto \log \beta$. In order to find the second order derivatives, there are two cases that we should deal with separately; first is when the derivative of Eq. (4.25) is taken with respect to the same variable $\log \beta_k$, in which case calculations lead to:

$$\begin{aligned} \frac{\partial^2 I(\boldsymbol{\tau})}{\partial \tau_k^2} \overset{[\tau \mapsto \log \beta]}{\equiv} \frac{\partial^2 L(\boldsymbol{\beta})}{\partial (\log \beta_k)^2} = \frac{\partial}{\partial \log \beta_k} \left[\beta_k \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} \right] &= \frac{\partial}{\partial \beta_k} \left[\beta_k \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} \right] \left(\frac{\partial \log \beta_k}{\partial \beta_k} \right)^{-1} \\ &= \beta_k^2 \frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_k^2} + \beta_k \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} \end{aligned} \tag{4.26}$$

It is also necessary to find the second order derivative when the variable indices are different; i.e. derivation with respect to $\log \beta_k$ when $k \neq j$:

$$\frac{\partial^2 I(\boldsymbol{\tau})}{\partial \tau_j \partial \tau_k} \overset{[\tau \mapsto \log \beta]}{\equiv} \frac{\partial^2 L(\boldsymbol{\beta})}{\partial \log \beta_j \partial \log \beta_k} = \frac{\partial}{\partial \log \beta_j} \left[\beta_k \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_k} \right] = \beta_k \beta_j \frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k} \tag{4.27}$$

These results show that in order to find the first and second order derivatives of $I(\boldsymbol{\tau})$,

we just need to find the corresponding derivatives of the original function $L(\boldsymbol{\beta})$, replace them into Eqs. (4.25)-(4.27), and then map every variable according to the mapping defined in Eq. (4.24).

We have already found the first derivative of $L(\boldsymbol{\beta})$ in Eq. (4.17). As for the second derivative when the derivative is taken with respect to the same variable, we have:

$$\frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_k^2} = \dot{L}_1(\beta_k) + \dot{L}_2(\boldsymbol{\beta}) \tag{4.28}$$

Here we have simplified the equations by using the functions introduced in Eq. (4.19). Also note that $\dot{L}_1$ and $\dot{L}_2$ are the derivatives of their corresponding functions[1]; that is to say:

$$\begin{aligned} \dot{L}_1(\beta_k) &= \sum_{i=1}^{n} \left(\psi_1(c_{ik} + \beta_k) - \psi_1(\beta_k)\right) \\ \dot{L}_2(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \left(\psi_1\left(\sum_{k=1}^{K} \beta_k\right) - \psi_1\left(\sum_{k=1}^{K} (c_{ik} + \beta_k)\right)\right) \end{aligned} \tag{4.29}$$

where ψ_1 is the trigamma function[2]. For the case where $k \neq j$, we are easily led to:

$$\frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k} = \dot{L}_2(\boldsymbol{\beta}) \tag{4.30}$$

We should now replace these equations into Eqs. (4.25)-(4.27) and map the variables according to Eq. (4.24). Let us denote the gradient of $I(\boldsymbol{\tau})$ by the vector $\mathbf{d}$, in other words:

$$\nabla I(\boldsymbol{\tau}) = \mathbf{d}(\boldsymbol{\tau}) = (d_1(\boldsymbol{\tau}), \ldots, d_K(\boldsymbol{\tau}))^T \tag{4.31}$$

We can easily find the entries of the vector $\mathbf{d}(\boldsymbol{\tau})$ by applying our mapping to Eq. (4.25), which leads us to:

$$d_k(\boldsymbol{\tau}) = \frac{\partial I(\boldsymbol{\tau})}{\partial \tau_k} = e^{\tau_k}\Big(L_1(\beta_k \mapsto e^{\tau_k}) + L_2(\boldsymbol{\beta} \mapsto e^{\boldsymbol{\tau}})\Big) \tag{4.32}$$

where we have defined $L_1(\beta_k \mapsto e^{\tau_k})$ and $L_2(\boldsymbol{\beta} \mapsto e^{\boldsymbol{\tau}})$ such that they represent their corresponding functions after we apply the the mappings $\beta_k \mapsto e^{\tau_k}$ and $\boldsymbol{\beta} \mapsto e^{\boldsymbol{\tau}}$, re-

[1] Throughout this thesis, we use Newton's notation for differentiation and place a dot over the function name to represent its derivative.

[2] The trigamma function is the derivative of the digamma function, in other words: $\psi_1(x) = \frac{d^2}{dx^2}\log\Gamma(x) = \frac{d}{dx}\psi(x)$.

spectively. With this representation, we can now write the second order derivatives as:

$$\frac{\partial^2 I(\boldsymbol{\tau})}{\partial \tau_k^2} = e^{2\tau_k}\Big(\dot{L}_1(\beta_k \mapsto e^{\tau_k}) + \dot{L}_2(\boldsymbol{\beta} \mapsto e^{\boldsymbol{\tau}})\Big) + d_k(\boldsymbol{\tau}) \tag{4.33}$$

$$\frac{\partial^2 I(\boldsymbol{\tau})}{\partial \tau_j \partial \tau_k} = e^{\tau_j} e^{\tau_k}\Big(\dot{L}_2(\boldsymbol{\beta} \mapsto e^{\boldsymbol{\tau}})\Big) \tag{4.34}$$

In order to simplify mathematical calculations, we write the Hessian matrix as a $K \times K$ matrix as:

$$\mathbf{H}I(\boldsymbol{\tau}) = \mathbf{Q} + \mathbf{u}\mathbf{u}^T z \tag{4.35}$$

where $\mathbf{Q}$ is a diagonal matrix and z is constant for all parameters, each defined by:

$$q_{jk} = \delta_k(j)\Big(e^{2\tau_k}\dot{L}_1(\beta_k \mapsto e^{\tau_k}) + d_k(\boldsymbol{\tau})\Big) \tag{4.36}$$

$$z = \dot{L}_2(\boldsymbol{\beta} \mapsto e^{\boldsymbol{\tau}}) \tag{4.37}$$

$$\mathbf{u} = (e^{\tau_1}, \ldots, e^{\tau_K})^T \tag{4.38}$$

To find the inverse of the Hessian matrix, as required by the Newton–Raphson iteration in Eq. (4.21), we make use of the Sherman-Morrison theorem [Sherman & Morrison, 1950; Press et al., 2007] which extremely simplifies the necessary calculations:

Theorem 4.1. *Suppose* $\mathbf{A}$ *is an invertible matrix, and* $\boldsymbol{v}_1, \boldsymbol{v}_2$ *are vectors with the constraint that* $\boldsymbol{1} + \boldsymbol{v}_2^T \mathbf{A}^{-1} \boldsymbol{v}_1 \neq \boldsymbol{0}$*. We have:*

$$(\mathbf{A} + \boldsymbol{v}_1 \boldsymbol{v}_2^T)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\boldsymbol{v}_1 \boldsymbol{v}_2^T \mathbf{A}^{-1}}{1 + \boldsymbol{v}_2^T \mathbf{A}^{-1} \boldsymbol{v}_1} \tag{4.39}$$

By letting $\mathbf{A} = \mathbf{Q}$, $\mathbf{v}_1 = \mathbf{u}$ and $\mathbf{v}_2^T = \mathbf{u}^T z$, Eq. (4.35) will have the same functional form required by Theorem 4.1. Furthermore, given that $\mathbf{Q}$ is a diagonal matrix, its inverse is also a diagonal matrix with diagonal entries equal to $1/q_{kk}$. With all this in mind, we can write the Newton–Raphson step for the mapped function $I(\boldsymbol{\tau})$ as:

$$[\mathbf{H}I(\boldsymbol{\tau})]^{-1} \nabla I(\boldsymbol{\tau}) = \left(\mathbf{Q} + \mathbf{u}\mathbf{u}^T z\right)^{-1} \mathbf{d} \tag{4.40}$$

$$= \mathbf{Q}^{-1}\mathbf{d} - \frac{\mathbf{Q}^{-1}\mathbf{u}\mathbf{u}^T\mathbf{Q}^{-1}\mathbf{d}}{1/z + \mathbf{u}^T\mathbf{Q}^{-1}\mathbf{u}} \tag{4.41}$$

After doing the matrix multiplications, we get to the following simplified representation for the Newton–Raphson step when updating the k'th element of the parameter

vector $\boldsymbol{\tau}$:

$$\tau_k^{\text{new}} = \tau_k - \Big(\left[\mathbf{H} I(\boldsymbol{\tau})\right]^{-1} \nabla I(\boldsymbol{\tau}) \Big)_k = \tau_k - \frac{d_k - e^{\tau_k} b}{q_{kk}} \tag{4.42}$$

where b is equal to:

$$b = \frac{\sum_{i=1}^{K} \frac{d_i e^{\tau_i}}{q_{ii}}}{\frac{1}{z} + \sum_{i=1}^{K} \frac{e^{2\tau_i}}{q_{ii}}} \tag{4.43}$$

The final step is to map Eqs. (4.42) and (4.43) back to the original setting, where we were dealing with the parameters of the Dirichlet distribution $\boldsymbol{\beta}$ using the mapping defined in Eq. (4.24):

$$\log \beta_k^{\text{new}} = \log \beta_k - \frac{\beta_k g_k - \beta_k b'}{q'_{kk}} \tag{4.44}$$

$$\beta_k^{\text{new}} = \beta_k \exp(-\beta_k \frac{g_k - b'}{q'_{kk}}) \tag{4.45}$$

where $\mathbf{g}$ was defined in Eq. (4.16), and b', q'_{ij} are the corresponding values for our mapping; i.e.:

$$q'_{jk} = \delta_k(j) \Big(\beta_k^2 \dot{L}_1(\beta_k) + \beta_k g_k(\boldsymbol{\beta}) \Big) \tag{4.46}$$

$$z' = \dot{L}_2(\boldsymbol{\beta}) \tag{4.47}$$

$$b' = \frac{\sum_{i=1}^{K} \frac{\beta_i^2 g_i}{q'_{ii}}}{\frac{1}{z'} + \sum_{i=1}^{K} \frac{\beta_i^2}{q'_{ii}}} \tag{4.48}$$

Direct calculation of each update term in Eq. (4.45) requires only simple additions and multiplications which saves computation time in contrast to inverting the Hessian matrix that takes $O(K^3)$ when we use the popular *Gauss Jordan elimination technique* [Strang, 2003]. In the next section, we will propose a second alternative approach to calculate the update term of the Newton–Raphson technique that satisfies the constraints imposed by the Dirichlet distribution by using a barrier function instead.

Newton–Raphson using a Logarithmic Barrier

Remember that our goal in this section is to solve an optimization problem which includes a number of inequality constraints. We can rewrite this problem in the following form:

$$\begin{aligned} &\text{maximize} && L(\beta_1, \ldots, \beta_K) = \log p(D|\boldsymbol{\beta}) \\ &\text{subject to} && \beta_k > 0 \qquad\qquad k = 1, \ldots, K \end{aligned} \tag{4.49}$$

In Section 4.1.2, we tried to address this problem by mapping each parameter to an auxiliary exponential variable with no constraints, which insures the positivity of our Dirichlet parameters. Here we will provide an alternative method to solve this problem by introducing a logarithmic barrier [Boyd & Vandenberghe, 2004].

The essence of this method is to make the inequality constraints implicit by introducing an *indicator function* for each parameter into the objective function:

$$\text{maximize } \log p(D|\boldsymbol{\beta}) + \sum_{k=1}^{K} I(\beta_k) \tag{4.50}$$

where the indicator function is defined in a way that it forces Eq. (4.50) to always return positive values; i.e.:

$$I(\beta_k) = \begin{cases} 0 & \beta_k > 0 \\ -\infty & \beta_k \leq 0 \end{cases} \tag{4.51}$$

The problem with Eq. (4.51) is that it is not differentiable and thus we cannot use it in our Newton–Raphson algorithm. However, we may circumvent this problem by approximating the indicator function by the function:

$$\hat{I}(\beta_k) = \frac{1}{t} \log \beta_k \tag{4.52}$$

This approximation has the property that returns $-\infty$ for negative values of β_k. The parameter $t > 0$ sets the accuracy of the approximation; as we increase its value, $\hat{I}(\beta_k)$ will become a more accurate approximation for the true function $I(\beta_k)$.

Substituting $\hat{I}(\beta_k)$ for $I(\beta_k)$ in Eq. (4.50) returns the following approximation for our optimization problem:

$$\text{maximize } L_t(\boldsymbol{\beta}) = \log p(D|\boldsymbol{\beta}) + \sum_{k=1}^{K} \frac{1}{t} \log(\beta_k) \tag{4.53}$$

which we can now apply the Newton–Raphson method as before to solve for its optimal values. To do this, we have to write down the gradient vector and Hessian matrix of $L_t(\boldsymbol{\beta})$ by finding its first and second order derivatives. Using the same definitions of Eq. (4.19), we find the first order derivative as:

$$g_k(\boldsymbol{\beta}) = \frac{\partial L_t(\boldsymbol{\beta})}{\partial \beta_k} = L_1(\beta_k) + L_2(\boldsymbol{\beta}) + \frac{1}{t\beta_k} \tag{4.54}$$

where $\mathbf{g}$ is the gradient of L_t; i.e. $\mathbf{g}(\boldsymbol{\beta}) = (g_1(\boldsymbol{\beta}), \ldots, g_K(\boldsymbol{\beta})) = \nabla L_t(\boldsymbol{\beta})$. For the second order derivatives, we obtain:

$$\frac{\partial^2 L_t(\boldsymbol{\beta})}{\partial \beta_k^2} = \dot{L}_1(\beta_k) + \dot{L}_2(\boldsymbol{\beta}) - \frac{1}{t\beta_k^2} \tag{4.55}$$

$$\frac{\partial^2 L_t(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k} = \dot{L}_2(\boldsymbol{\beta}) \qquad j \neq k \tag{4.56}$$

As before, we may now write the Hessian matrix as:

$$\mathbf{H}L_t(\boldsymbol{\beta}) = \mathbf{Q} + \mathbf{u}\mathbf{u}^T z \tag{4.57}$$

where $\mathbf{u} = (1, \ldots, 1)^T$ is the unit vector, and Q is a diagonal matrix with entries equal to:

$$q_{jk} = \delta_k(j)\left(\dot{L}_1(\beta_k) - \frac{1}{t\beta_k^2}\right) \tag{4.58}$$

$$z = \dot{L}_2(\boldsymbol{\beta}) \tag{4.59}$$

Once again we use Theorem 4.1 and find the inverse Hessian matrix. After completing the computations, we get the following Newton–Raphson step for our approximate optimization:

$$\beta_k^{new} = \beta_k - \left([\mathbf{H}L_t(\boldsymbol{\beta})]^{-1} \nabla L_t(\boldsymbol{\beta}) \right)_k \tag{4.60}$$

$$= \beta_k - \left(\frac{g_k - b}{q_{kk}} \right) \tag{4.61}$$

where b is the same for all parameters and we can thus calculate it once as:

$$b = \frac{\sum_{i=1}^{K} \frac{g_i}{q_{ii}}}{\frac{1}{z} + \sum_{i=1}^{K} \frac{1}{q_{ii}}} \tag{4.62}$$

The Newton–Raphson step obtained in Eq. (4.61) is just an approximation of the original optimization problem expressed in Eq. (4.50). In this approach, as we increase the parameter t, $\hat{I}(\beta_k)$ will look more like $I(\beta_k)$. However, the problem with taking a large value for t is the difficulty we face when optimizing using the Newton–Raphson method [Boyd & Vandenberghe, 2004]. We can circumvent this problem by finding the associated *central path*; where we solve a sequence of optimization problems, starting each Newton–Raphson step at the solution of the problem for the previous value of t [Boyd & Vandenberghe, 2004].

We conclude our remarks on estimation for finite dimensional mixture models and now turn to their infinite dimensional variants. In the following section, we will cover some of the fundamental approaches to estimating the hyperparameters in mixture models with potentially infinite number of components.

4.2 Infinite Mixture Hyperparameter Estimation

Consider the infinite mixture model described in section 3.3.2. From the definition of the model, it is obvious that the concentration parameter of the Dirichlet process, namely γ, plays an important role in our modelling because as we mentioned earlier, the expected number of components is directly related to the value of γ. As Doss [2008] noted, this parameter is the most difficult to estimate or defend as a fixed value. Escobar & West [1995] proposed a Gibbs sampling algorithm based on West [1992] where γ is assumed to have a Gamma distribution prior Gamma(a, b) with shape $a > 0$ and scale $b > 0$. In their clever approach, they reduce the estimation to successive draws from Beta and Gamma distributions. However, the problem now switches to estimating reasonable values for a and b.

Some authors try to find the best values for the Gamma prior by varying a and b over different values (see, e.g. [Kottas, 2006]), while others follow West & Escobar [1993] and choose a and b such that the prior has a reasonable mean and a relatively low variance [Mukherjee et al., 2007; Gelfand et al., 2005]

Here we will discuss two of the most prevalent hyperparameter estimation techniques for the Dirichlet process mixture model; first a blend of deterministic and stochastic approximation methods, proposed in Dorazio [2009], and second a simple Gibbs sampling approximation based on the maximum likelihood estimate of γ as introduced in McAuliffe et al. [2006].

4.2.1 Estimation Using Prior Information

Suppose we draw a random distribution from a Dirichlet process $G \sim DP(\gamma, H)$ and then sample n observations from our discrete distribution G; i.e. $\mathbf{y} = (y_1, \ldots, y_n) \sim G$. From Theorem 3.3 we know that these observations will always take $K \leq n$ values. Also recall from Eq. (3.18) in Section 3.2.5 that the number of distinct values among our observations will have the following probability distribution:

$$p(k|n, \gamma) = \frac{\Gamma(\gamma)}{\Gamma(\gamma + n)} s(n, k) \gamma^k$$

where $s(n, k)$ are the unsigned Stirling numbers of the first kind. Dorazio [2009] elaborates on this distribution and, following West [1992], assumes a Gamma(a, b) prior for the concentration parameter γ. If we find the marginal distribution of k by integrating γ out, we obtain:

$$p(k|n, a, b) = \int_0^\infty p(k|n, \gamma) p(\gamma|a, b) d\gamma \tag{4.63}$$

$$= \frac{b^a s(n, k)}{\Gamma(a)} \int_0^\infty \frac{\gamma^{k+a-1} exp(-b\gamma)\Gamma(\gamma)}{\Gamma(\gamma + n)} d\gamma \tag{4.64}$$

The integral in Eq. (4.64) cannot be calculated analytically and we must resort to numerical methods to evaluate the integral for fixed a, b and n.

Suppose we have prior information about the number of components, which we represent using the probability distribution $q(k)$. What we are looking for is then to make $p(k|n, a, b)$ close to our prior distribution $q(k)$, and one way to do this similarity comparison is to calculate the Kullback-Leibler (KL) divergence between these two distributions. Since k is a discrete value between 1 and n, we should replace the integral in Eq. (2.40) with a summation:

$$\mathrm{KL}(q(k)||p(k|n, a, b)) = \sum_{k=1}^{n} q(k) \log \left(\frac{q(k)}{p(k|n, a, b)} \right) \tag{4.65}$$

In the absence of any prior information, a wise choice for $q(k)$ would be to give equal probability to all possible values; i.e. $q(k) = 1/n$ for $k = 1, \ldots, n$. Under this assumption, The KL divergence then becomes:

$$\mathrm{KL}(q(k)||p(k|n, a, b)) = -\log n - \frac{1}{n} \sum_{k=1}^{n} \log p(k|n, a, b) \tag{4.66}$$

The goal is to find the values of a and b that minimize the KL divergence, thus making

$p(k|n,a,b)$ as close as possible to our prior opinion of the number of distinct components. After replacing $p(k|n,a,b)$ in Eq. (4.66) and removing the terms that do not depend on a or b, Eq. (4.66) amounts to maximising the following objective function:

$$\text{maximize} \quad a\log b - \log\Gamma(a) + \frac{1}{n}\sum_{k=1}^{n}\left(\int_0^\infty \frac{\gamma^{k+a-1}exp(-b\gamma)\Gamma(\gamma)}{\Gamma(\gamma+n)}d\gamma\right) \tag{4.67}$$

Calculation of the first and second terms in Eq. (4.67) is straightforward, however numerical evaluation of the last term is computationally demanding, especially when we are dealing with a large number of observations (as with the Dihanna corpus where we have thousands of utterances). Dorazio [2009] calculates the optimal values of the Gamma distribution under a uniform $q(k)$ for sample sizes ranging from $n = 5$ to 50.

Now assume that we have found the optimal values a and b and have thus formed the prior $p(\gamma|a,b)$. In order to make it possible to implement a Gibbs sampling algorithm, we should identify the conditional posterior distributions that determine our variables. For the hyperparameter γ, the posterior density depends only on the realized number of components, the number of observations and our prior as follows:

$$p(\gamma|k,n,a,b) \propto p(k|n,\gamma)p(\gamma|a,b) \tag{4.68}$$

where the likelihood function is given in Eq. (4.63). West [1992] has proposed a clever approach for calculating Eq. (4.68) by identifying an auxiliary Beta-distributed variable x which, after simplification, results in drawing samples from a mixture of Gamma distributions. In other words:

$$\begin{aligned}\gamma|x,k,a,b \quad \sim \quad & w_x\text{Gamma}(a+k, b-\log(x)) \\ & +(1-w_x)\text{Gamma}\left(a+k-1, b-\log(x)\right)\end{aligned} \tag{4.69}$$

with weights w_x defined by:

$$\frac{w_x}{(1-w_x)} = \frac{(a+k-1)}{n(b-\log(x))} \tag{4.70}$$

and the variable x drawn from:

$$x|\gamma,k \sim \text{Beta}(\gamma+1, n) \tag{4.71}$$

We can now successively draw samples of our hyperparameter γ by iterating through a series of Gibbs sampling steps. First we should draw samples from the Dirichlet process

mixture model using an algorithm such as Algorithm 2 explained in Section 3.3.3. Note that no matter what sampling algorithm used, the knowledge of $\mathbf{z}$ (the vector of indicator variables) or $\boldsymbol{\Theta}$ (the matrix of parameters of each component) is equivalent to knowledge of k. Given k, we draw a new value of γ by first sampling an x value from Eq. (4.71) and then sampling γ from the mixture of Gammas in Eq. (4.69) conditional on the k and x values obtained in the current step. The entire sampling scheme is shown in Algorithm 3.

Algorithm 3: Gibbs sampling of the DPMM concentration parameter γ

Choose a starting point $\mathbf{z}^0 = (\mathbf{z}_1^0, \mathbf{z}_2^0, \ldots, \mathbf{z}_n^0)$;
Generate new samples as follows
for $j+1 = 1$ **to** T **do**
  for $i = 1$ **to** n **do**
    Sample z_i^{j+1} from $p(z_i | z_{(-i)}^j, y_i, \gamma, H)$ as described in Algorithm 2
  end
  k^{j+1} = number of distinct values in the vector $\mathbf{z}^{j+1}$
  Sample $x^{j+1} \sim \text{Beta}(\gamma^j + 1, n)$
  begin
    $\gamma^{j+1} \sim \text{Gamma}(a + k^{j+1}, b - \log(x^{j+1}))$ with probability w_x^{j+1}
    $\gamma^{j+1} \sim \text{Gamma}\left(a + k^{j+1} - 1, b - \log(x^{j+1})\right)$ with probability $(1 - w_x^{j+1})$
  end
end

We will conclude our discussion on this method by noting an important fact regarding the Gamma prior on γ. In fact, we could have used any other distribution to specify our uncertainty in the prior. However, the main reason for choosing a Gamma distribution is that it will lead to the clever sampling scheme of West [1992].

Also note that Algorithm 3 does have its own drawbacks. As we discussed earlier, calculating the optimal values for the Gamma distribution prior is not computationally efficient when the number of samples is large. In the following section, we will discuss another sampling algorithm which is based on the maximum likelihood estimate of the hyperparameter γ.

4.2.2 Estimation Based on MLE

In Section 3.2.5, we showed that in a Dirichlet process, the expected number of unique observations in a sequence of n samples is given by:

$$E(K|n,\gamma) = \sum_{i=1}^{n} \frac{\gamma}{\gamma + i - 1}$$

The fundamental result that will be used in this estimation technique is Theorem 4 in Liu [1996]. A formal statement of this theorem is given below:

Theorem 4.2. *The maximum likelihood estimate (MLE) of the hyperparameter γ of a Dirichlet process DP(γ, H) satisfies the following stationary condition:*

$$\begin{aligned} E(K|n,\gamma,y_1,\ldots,y_n) &= E(K|n,\gamma) \\ &= \sum_{i=1}^{n} \frac{\gamma}{\gamma + i - 1} \end{aligned} \tag{4.72}$$

Liu [1996] then proceeds to give a method based on *sequential imputations*, although he notes that it is possible to do approximation using the Gibbs sampler. McAuliffe et al. [2006] propose the Gibbs sampling method that is based on Theorem 4.2. In this approach, we repeatedly draw samples from our Dirichlet process mixture model using a sampling scheme such as the one explained in Section 3.3.3. We then average over the number of distinct components found in each step and equate it with Eq. (4.72) to find the new value of γ. The algorithm is described in detail below.
McAuliffe et al. [2006] noticed that the initial configuration of γ does not affect the final result. In fact, they have initialized γ with a plausible set of values, all of which have converged to nearly the same value. They use this method in their model of velocity measurements for 82 galaxies from diverse sections of the Corona Borealis region. This method has also been used by Dorazio et al. [2008] in approximating the hyperparameter of the DPMM which they use to model the sources of heterogeneity in animal abundance.

4.3 Summary

This chapter covered some basic methods for estimating the hyperparameters of finite and infinite dimensional mixture models. For the former model, previous attempts have

Algorithm 4: Gibbs sampling of the DPMM concentration parameter γ using MLE

Choose a starting point $\mathbf{z}^0 = (\mathbf{z}_1^0, \mathbf{z}_2^0, \ldots, \mathbf{z}_n^0)$;
Generate new samples as follows
for $j+1 = 1$ **to** T **do**
  for $r+1 = 1$ **to** R **do**
    for $i = 1$ **to** n **do**
      Sample z_i^{r+1} from $p(z_i | z_{(-i)}^r, y_i, \gamma^j, H)$ as described in Algorithm 2
    end
    k^{r+1} = number of distinct values in the vector $\mathbf{z}^{r+1}$
  end
  $\hat{k} = \frac{1}{R} \sum_{r=1}^{R} k^r$
  Find γ^{j+1} that satisfies $\hat{k} = \sum_{l=1}^{n} \frac{\gamma^{j+1}}{\gamma^{j+1} + l - 1}$
end

neglected the fact that the parameters of the Dirichlet distribution must always remain positive, so we presented two novel approaches based on constraint optimization. The first approach used an exponential mapping to enforce the constraints, whereas in the second one we approximated the constraints using a logarithmic barrier function.

In the next part, we discussed two methods for estimating hyperparameters in the Dirichlet process mixture model. The first approach was a deterministic approximation which tried to approach the true function with a simpler and easier to evaluate function, and the second approach was based on Gibbs sampling.

In the experiments that we will describe in the following chapter, we make use of the estimation techniques presented here to find the appropriate parameters of our model. As we will explain in more detail, the whole scheme is a DPMM with (potentially) infinite number of components, where in each component the members are governed by a Dirichlet-Multinomial distribution.

Chapter 5

Application in Automatic Dialogue Act Classification

One of the greatest challenges in natural language processing is the recognition of the speaker's intention based on the utterances that he has made during the conversation. This is done by identifying the underlying semantic units of the dialogue which are usually coded in terms of Dialogue Acts. Manual annotation of DA's is both time consuming and expensive, therefore there is a huge interest in systems which are able to automatically annotate dialogue corpora. In this section, we first present an overview of existing automatic classification techniques, and then introduce a technique based on the Dirichlet process mixture model explained in the previous chapters. We conclude the section with a series of experiments on classifying the DIHANA corpus using DPMM's and verifying results with the actual annotations in the corpus.

5.1 Dialogue Acts; an Introduction

Dialogue Acts [Bunt, 1994; Stolcke et al., 2000; Webb, 2010] are one of the most important mechanisms for dialogue analysis. These units are very similar to the concept of *speech act* introduced by Austin [1962] and further developed by Searle [1969]. In his influential work, Austin explains how utterances are more than just descriptions of states of affairs; that they can actually perform actions and change mental states of the participants. For example, by saying 'bring me the newspaper', an act is performed and something special has been done.

Utterance	Dialogue Act
Do you have any special training?	Yes-No-Question
Uh-huh	Acknowledge
I think it's great	Statement-Opinion
I can imagine	Appreciation

Table 5.1: Example of a typical dialogue act

Austin [1962] distinguished between three types of actions we can do by making an utternace; a *locutionary act* is the act of saying something, like referring to or naming an object. A *perlocutionary act* is the act performed by saying something, and an *illocutionary act* which is the act performed in saying something. In the same example given above, just by saying 'bring me the newspaper' we have performed a locutionary act. Moreover, I have made another action in ordering you to do something for me (illocutionary act), and if you listen to me and bring me the newspaper, I have successfully done what I intended to do (perlocutionary act).

Searle [1969] refined Austin's work on illocutionary acts by providing necessary and sufficient conditions for the act to be performed. He also proposed a taxonomy of speech acts based on the purpose of the acts and the dimensions along which they can vary [Traum, 1999].

Whereas a full treatment of taxonomies and properties of speech acts provides insights into the study of linguistics, mostly the practical aspects of these acts are of interest within the computation view of linguistics. In the development of spoken dialogue systems, it is more important to consider the conversational roles such acts can perform. For example, if an automatic agent asks a question and the user replies with another question, it is probably in order to clarify the original question [Webb, 2010]. These are called *Dialogue Acts* [Bunt, 1994], or in some contexts *Conversational Moves* or *Dialogue Moves* [Power, 1979], or *Adjacency Pair Parts* [Schegloff, 1968; Sacks et al., 1974]. Therefore, while speech acts deal with the intentions and actions of saying utterances, dialogue acts are annotations that give functional roles to the utterances of a dialogue.

Table 5.1 shows what sort of linguistic structure we mean when we talk about dialogue acts. In fact, we want to know what role each utterance plays in the dialogue; whether it functions as a question, an acknowledgement, a confirmation, or any other functional role. However, there is no accepted universal tagging system for labelling the utterances of a dialogue. Therefore, one might wind up with different labels when using different

annotation schemes on the same corpus. In section 5.1.2, we will review some of the DA annotation schemes developed in several labelling projects and which are common among the spoken dialogue community. But before that, we will overview some of the most important uses of dialogue acts in machine learning.

5.1.1 Applications of Dialogue Acts

Probably the most intuitive application of DA's are their use in dialogue systems. A dialogue system can be defined as a system which is capable of conversing with a human in a coherent structure. In such systems, identification of dialogue acts can help recognize the intentions of the speaker, which in turn allows the system to make more appropriate responses. There is a very broad literature on these types of applications. Examples are Tamarit et al. [2011] or Prasad & Walker [2002] where each try to annotate a different corpus for building a dialogue manager for a specific task. Fernández et al. [2005] investigated the use of dialogue acts for understanding special linguistic units called *non-sentential utterances* or NSU's, which are fragmentary utterances that convey a full sentential meaning. Even more general, Lendvai et al. [2003] used DA classification techniques for *shallow interpretation*, which gives a more general understanding by combining both semantic and pragmatic aspects of the utterance.

Another venue for applying dialogue acts in practice lies in the area of automatic machine translation. As noted by Fishel [2006], correctly labelled utterances can help reduce the ambiguities when a sentence might have more than one meaning. For example, despite its grammatical structure, an utterance such as 'would you mind opening the window' should not be considered as a question, but rather as a request. One typical example of such a project is the VerbMobil project [Wahlster, 1993; Jekat et al., 1995; Wahlster, 2000]. Verbmobil is a bidirectional translation system for spontaneous dialogues in mobile situations which can handle dialogues in either German, English or Japanese. Other examples include Levin et al. [2003] where the authors used what they call *domain actions* in order to label the utterances. These domain actions are in fact a combination of domain independent dialogue acts followed by domain dependent concepts. Lee et al. [1997] also proposed a dialogue model based on dialogue acts for Korean-English machine translation.

There has also been a wide literature on the use of dialogue acts in social meaning and its detection. DA structures have been used to annotate emotions such as annoyance or frustration [Ang et al., 2002; Craggs & Wood, 2003; Liscombe et al., 2005]. In another not so different context, Rosenberg & Hirschberg [2005] used a set of 26 tags such as passionate, enthusiastic, or desperate to annotate speaker characteristics. Their

goal is to understand how do charismatic leaders attract and retain their followers. In more social phenomena such as speed dating, Jurafsky et al. [2009]; Ranganath et al. [2009] used dialogue acts to label speakers as flirtatious, awkward or friendly. There has also been efforts to tag speaker depression or stress [Pennebaker et al., 2007; Rude et al., 2004] or personality traits such as extroversion, agreeableness or conscientiousness [Mairesse & Walker, 2008; Mairesse et al., 2007] using dialogue act structures.

While this is not a comprehensive list of the various applications dialogue acts have in practice, it should give a clear picture of how they could be used in different areas of research. However, their application requires answers to two important questions. First, how should we label the utterances in a given dialogue, are there accepted frameworks for labelling in different domains? And second, given that manual annotation is cumbersome, what are the approaches that we can use to automate this process? We begin with the former question and describe some of the schemes that have been devised for dialogue act annotation.

5.1.2 Dialogue Act Annotation Schemes

In the past few years, the use of corpus based approaches to natural language processing have gained significant importance and large corpora for different applications are being collected all over the world. However, unless we use a completely unsupervised method for training and testing our learning algorithms, these corpora need to be annotated based on the task they are to perform. For this we need annotation schemes which correspond to the task, domain, and linguistic phenomena related to that specific project.

Dialogue act annotation schemes must make a compromise between two factors [Fishel, 2006]. First, the definitions they use for the tags must be clear enough so that it would reduce any possible disagreement between human taggers. Second, it is preferable to define the tags in such a way that they could be easily generalized to problems in other domains as well. However, a balance between these two factors has proven to be quite a difficult task.

There seems to be little consensus on a standard DA tag set for all domains. One key factor which impedes such a consensus even in domain related corpora is the granularity of approaches used for this task [Webb, 2010]. For example, consider the following utterance from the DIHANA corpus:

U:I want to know the fare for the train leaving at 7:45.

The approach used in the DIHANA project attempts to concatenate the semantic and functional information into a single tag and assigns the label <Question-Fare-Departure Hour> to the utterance. However, one can also adopt a higher level approach by leaving out the semantic analysis and labelling the utterance as a simple question. Whereas finer labels may be easier to use for the specific task they are designed for, but their generality is limited to models related to their specific application.

Many annotation schemes have been proposed in the past few years, where each is suitable for a specific task. Some like the Dialogue Act Mark-up in Several Layers, DAMSL, have been defined with the goal of providing a domain independent structure for annotating a range of dialogues for many different purposes [Core & Allen, 1997]. This corpus and its variants have been used to annotate many dialogue corpora. One widely used variant of this scheme was designed for the Switchboard corpus and is called SWBD-DAMSL [Jurafsky et al., 1997]. This markup language contains roughly 80% of the original tags, with some of the DAMSL categories subdivided and some new categories added to the original scheme.

Table 5.2 shows a basic overview of some of the most prevalent annotation schemes available, along with the corpora they are associated with. Some of these corpora are not widely used and are associated with a small number of dialogue projects.

The DIHANA corpus, which we have used for our experiments, is based on the *Interchange Format* (IF) defined in the C-STAR project [Fukada et al., 1998]. This scheme is a three level representation based on domain actions, which consists of *speech acts* plus domain specific *concepts*. The speech act captures the intention of the speaker and the concepts capture the information focus of the utterance. Along these, there is an *argument* at the end of each label which represent the specific content of the utterance. For example, consider the utterance 'a double room costs 150 dollars a night'. This phrase consists of the speech act <give-information> with two main concepts <+price> and <+room>. A typical dialogue act for this utterance could thus be <give-information + price(price=150) + room(room=double)> [Levin et al., 2000]. We will explain the labelling scheme used for the DIHANA corpus with more detail in Section 5.3.1.

Before proceeding to the next section, it should be noted that the switchboard corpus along with the SWBD-DAMSL annotation scheme was also a candidate for our experiments. However, since the switchboard corpus is larger than DIHANA, and the range of topics discussed in the corpus is wide and domain independent, we finally decided to use the DIHANA corpus for this phase and leave the more challenging switchboard corpus for future work.

Annotation Scheme	# of Dialogues	# of Labels	Underlying Task	Language
MAPTASK	128	12	Cooperative game to draw a route	English
VERBMOBIL	1172	43	Automatic machine translation	English,German,Japanese
DAMSL	-	4 dimensions	Domain independent	English
SWBD-DAMSL	1155	226	Domain independent	English
ICSI-MRDA	75	39	Multi-party meetings	English
C-STAR	230	26	Travel planning	English,Korean,Italian,Japanese
AMITIES	1000	259	Information seeking or transaction	English
AMI	100 hours	15	Multi-modal meeting room	English
Spanish CallHome	120	232	Friendly telephone calls	Spanish
Basurde	227	15	Information on trains	Spanish
DIT++	-	11 dimensions	Domain independent	English
DIHANA	900	72	Information on train times and fares	Spanish

Table 5.2: General information on some of the widely used annotation schemes, along with information about their associated dialogue corpus when available.

	Predicted Class = C_1	Predicted Class = C_2
True Class = C_1	TP: True Positive	FN: False Negative
True Class = C_2	FP: False Positive	TN: True Negative

Table 5.3: A binary confusion matrix

5.2 Automatic Dialogue Act Recognition Techniques

In what follows, we will examine several successful techniques which have been applied to automatic DA recognition. In order to compare the performance of each technique, an evaluation metric must be adopted. However, in contrast to other areas of machine learning, there is no public ground for evaluation in this area [Webb, 2010].

There are two commonly used methods for evaluating the results of automatic annotation methods. The first obvious way is to calculate the percentage of correctly classified utterances. This is the basic method we will use to compare performances in this section. The second way for measuring classification error is based on the *confusion matrix*. This is a $K \times K$ matrix with its (i, j)'th entry equal to the number of instances which belong to class C_i but are assigned to class C_j [Alpaydin, 2004]. The error rate is then defined as:

$$\text{error rate} = \frac{|FN| + |FP|}{N} \tag{5.1}$$

with $N = |TP| + |FP| + |TN| + |TP|$ the total number of instances. Table 5.3 helps better understand these concepts for the case with only two classes.

In the following sections, we will overview some of the most successful classification techniques used for dialogue acts. These methods can be categorised in several ways; either by the algorithm they use, the corpus on which they run their tests, or the range of features utilised. Here we adopt the first approach and in each case, after giving an overview of the method, summarize the results obtained on different corpora using that specific technique. We begin our analysis with methods based on sequence models.

5.2.1 N-gram Models

Perhaps the simplest method for predicting a DA is the use of sequence n-gram models. In this approach, the label for the next utterance is determined by the label of the k preceding utterances. In other words, the DA label chosen such that:

$$d_n = \underset{d}{\operatorname{argmax}}\, p(d|d_{n-1}, \ldots, d_{n-k+1}) \tag{5.2}$$

However, this method requires large annotated corpora to calculate the sequence probabilities. The conditional probabilities of Eq. (5.2) are then derived by counting the number of times the sequence $(d_{n-k+1}, \ldots, d_n)$ appears in the corpus, divided by the number of occurrences of $(d_{n-k+1}, \ldots, d_{n-1})$ and repeating this for all possible n. The value of k is normally taken to be equal to 2 or 3, larger values are only plausible when sequences of longer dependencies are known to exist.

Nagata & Morimoto [1994] first used this method on a small dataset with a small number of DA tags, and achieved 61.7% accuracy. They also evaluated their algorithm for the case where each utterance could have more than one correct label. Their results demonstrate a 77.5% accuracy using two tags and 85.1% when using three tags.

In another project, Reithinger & Maier [1995] used the same model over the VERBMOBIL corpus with only the top 18 DA tags (see table 5.2). By using only a single tag, they achieved 40.25% accuracy, while two or three tags increased the accuracy rate to 59.62% and 71.93% respectively.

Many authors, like Core & Allen [1997] found that using the n-gram model in conjunction with other approaches would yield better performances than using it alone. Most of the methods described below use some sort of combination in their approaches.

5.2.2 Hidden Markov Models

In this approach, we assume that the utterances along with the unobserved dialogue act labels form a finite *Hidden Markov Model* (HMM). An HMM is modelled as a set of unobserved discrete state variables and a set of observations generated by each of these states. Here we don't observe the true states, we just have access to the observations they produce. The parameters of this models are of two types, *transition probabilities* which control the transition between the unobserved states, and the *emission probabilities* which determine the probability of having one of the observations, given that we

are in a specific state. These parameters are estimated from the training corpus. For an excellent description on HMM's, refer to Rabiner [1989].

When These models are applied to DA classification, it is assumed that the utterances are the observed outputs, and the dialogue acts are the unobserved states that produce these utterances. In their influential article, Stolcke et al. [2000] apply HMM models to words in individual utterances, and report a 54.3% accuracy. However, they noted that by combining the HMM based model with n-grams of two and three lengths, the accuracy increases to 68.2% and 71% respectively. This shows how important dialogue context could be in understanding the current DA, and that immediate context provides a good deal of information [Webb, 2010].

Following their previous work, Reithinger & Klesen [1997] used the top 18 acts of the VERBMOBIL corpus and try to model the whole dataset using the HMM model described above. Their results show a 74.7% accuracy for the English data, and a 67.18% accuracy when applied to the German data. According to Reithinger & Klesen [1997], this difference is mainly due to the fixed word order of English, compared to the relatively free word order for German.

5.2.3 Bayesian Approaches

There are several Bayesian methods available which have proven to be successful in automatic DA classification. Among all, the *naive Bayes classifier* is probably the simplest Bayesian Approach. The basic idea behind this classifier is to maximize the probability of the DA tag given a set of features. However, a strong independence assumption among the features $f_1, f_2, \ldots f_m$ is assumed. In other words, we are looking for the class $\hat{c}$ such that:

$$\begin{aligned} \hat{c} &= \underset{c}{\operatorname{argmax}}\, p(c|f_1, \ldots, f_m) && (5.3) \\ &= \underset{c}{\operatorname{argmax}}\, p(c)p(f_1, \ldots, f_m|c) \\ &= \underset{c}{\operatorname{argmax}}\, p(c) \prod_i p(f_i|c) && (5.4) \end{aligned}$$

where the last line follows from the conditional independence of the features. Based on how we define our features, the components of Eq. (5.4) can be easily computed from the training corpus. Grau et al. [2004] used a bag of words[1] approach to form the features of clusters. They achieved a result of 60% on the SWBD-DAMSL corpus.

[1]The bag of words is a simplified model where each utterance, text, etc is regarded as an unordered collection of words, without considering any of the linguistic components of the source.

Using a modified version of the classifier, Ivanovic [2005] neglected the DA probabilities $p(c)$ in Eq. (5.4) and by applying the method to only a specific subset of the corpus, he obtained an 80% precision.

Many researchers have tried to question the strong assumption that the features are independent, and have proposed more advanced models based on *Bayesian Networks*. In this approach, the features are no longer independent from one another, instead, their dependencies are specified by acyclic directed graphs. So, for example, f_3 might depend on f_1 and f_4, but not on f_2. The class c is also regarded as some sort of feature which both depends and influences the other features. As before, the conditional probabilities are usually easy to calculate from the training corpus.

Keizer et al. [2002] used variants of this method on two different corpora. In their first experiment, they identified three features: *sentence type*, *subject type* and *sentence punctuation*. They produce a small Bayesian network and apply it to the Schisma corpus with tags very similar to the DAMSL scheme, and get 43.5% accuracy. In a second experiment, they create their own simple dialogue corpus annotated with the same tag set. This time, they use 13 features related to various characteristics of the utterances. Their final results demonstrate a 83% precision on this corpus.

5.2.4 Memory Based Learning

Memory Based Learning (MBL) is based on the intuitive hypothesis that humans handle situations by comparing them with previous situations. Not only when we make decisions, but also when we try to identify objects we make use of similarities with our understanding of that object. In a similar manner, in MBL we store some reference values and compare new unseen samples with these stored samples.

There are two questions we should answer when we adopt this approach. First, how should we compare values with those in the reference set? One widely used technique is the k-nearest neighbours (k-nn) classification, where we find the k closest values among the reference set to the unseen sample, and assign the label based on a majority voting. For this means, a simple yet powerful way to find the distance between pairs is to find the Euclidean distance.

The second question we should consider is the way we store the training samples. One common approach is to reduce them to a set of features, and save these features as classification information. By using a Euclidean distance metric, we thus calculate the distance between the features of an unseen sample with those in the reference set. The

algorithm described above is also called the IB1 algorithm [Daelemans et al., 2007] which is used in the Tilburg Memory Based Learner (TiMBL) software.

Levin et al. [2003] used the IB1 algorithm with 1 nearest neighbour in the NESPOLE machine translation project. Their approach assumes utterances are bags of words, but instead of word features, they use binary features indicating the presence or absence of labels from grammars. The result is a 69.82% accuracy. Lendvai et al. [2003] also used the TiMBL system on a corpus of 3,738 interactions with a Dutch train time table system, annotated with 94 different DA's, and achieved 73.5% precision. MBL has also been applied to the domain independent Switchboard corpus, using a 3-nn classification algorithm, and the reported percentage of correct classifications was 72.32% [Rotaru, 2002].

5.2.5 Decision Trees

A *Decision tree* has a tree structure where at each node, a condition is evaluated. The branch with which the condition holds specifies the sub-tree we should traverse. So by beginning from the root, we traverse through the tree based on the conditions on each arc. In our context, the input to each decision tree is an utterance, and each of the leaves of the graph specify a DA tag.

Compared to other methods, this approach has the basic advantage that it is much easier to understand by human analysts. They are also a popular method of classification in the whole machine learning literature due to their great performance. However, constructing a Decision tree could be difficult for large problems.

Using the same features as described previously, Levin et al. [2003] applied a decision tree to the same corpus and obtained 70.41% accuracy, which is almost the same performance as the MLB classifier. Verbree et al. [2006] also used a variant of the decision trees and apply them to three corpora, the ICSI-MRDA corpus, the Switchboard corpus and initial work on the AMI corpus. The set of features they used contained five categories; the presence or absence of the question mark (?) and the word *or*, the length of each segment, bi-gram of the previous two labels, and the presence of bi-grams, tri-grams and quadri-grams in utterances of both words and part of speech tags. With these features, they obtain a 89.27% precision on the ICSI-MRDA corpus which is the best result reported on this corpus, 65.68% on the Switchboard corpus and 59.76 on the AMI corpus.

5.2.6 Multilayer Perceptron

the *Multilayer Perceptron* is the most widely used neural network which is based on the biological working of the neurons inside human brains. These constructs are composed of computational units like neurons with input and output connections. Each connection has a weight which means that the input is multiplied by this number. The overall model consists of an input layer with one neuron per feature, a number of hidden layers, and an output layer which returns the vector of results.

In this model, each neuron has a nonlinear activation function which finds the weight of the output connections based on the input signals. This function is modeled in several ways, but must always be normalizable and differentiable [Alpaydin, 2004]. However, the most frequently used activation function is a sigmoid[1].

Sanchis & Castro [2002] used multilayer perceptrons on a tightly defined domain. Their corpus consisted of 215 Spanish dialogues in the train time table domain, annotated with 11 DA tags. They achieved a 92.54% of classification accuracy using transcribed dialogues.

Wright [1998] compared a range of classification algorithms on the MAPTASK corpus and get almost similar performance in all of them. By combining the algorithms with a quadri-gram DA progression, their HMM algorithm achieved 64% accuracy, their decision tree achieves 63% and their multilayer perceptron achieved a 62% precision. Levin et al. [2003] also applied the multilayer perceptron to the same corpus they used for the decision tree algorithm and the MLB, and found the perceptron to produce the best result among all algorithms, with a 71.52% accuracy.

We conclude our discussion on dialogue act recognition techniques with an important remark. In all the projects mentioned above, care must be taken when comparing results, even when the projects are performed on the same corpus. As noted by [Webb, 2010], there are several issues that must become clear before any comparison is carried out. These include information about the tag set used and the tags left out, the preprocessing of the corpus, the use of orthographic information, or even information about the splits used for training and testing purposes. However, they do show interesting facts about which algorithms are more suited to which problems.

What we presented here was just a brief overview of the previous works in automatic

[1] A sigmoid function is a function that produces a sigmoid curve, i.e. has a 'S' like shape. When detailed description is not present, it normally refers to the logistic function defined by $\frac{1}{1+e^{-t}}$

DA classification and is far from a complete discussion in the domain. Interested readers may refer to [Webb, 2010] or Fishel [2006] for more details on these approaches. In the remainder of this chapter, we will introduce a new approach to DA classification based on the concept of Dirichlet process explained in the previous chapters.

5.3 Experimental Results

In all of the methods presented in Section 5.2, we start with a predefined number of dialogue acts and after the training phase is over, the algorithm tries to assign utterances to one of these labels. The problem with this approach is that we must know how many dialogue acts we are going to have before running the algorithm; in other words, we must have previously labelled the corpus in order to be able to train our algorithm. The DA recognition algorithms lack to ability to learn the number of distinct classes themselves.

In an attempt to overcome this subtlety, we try to utilise the concept of the Dirichlet process we fully explained in Chapter 3. Remember we showed that a great characteristic of nonparametric Bayesian approaches was in their flexibility; that through the Chinese Restaurant metaphor, we can let the data to speak for themselves and determine how many clusters are necessary for classifying them. We begin with an overview of the DIHANA corpus that we used for our algorithm, along with the annotation scheme associated to the corpus, and present our experimental results in the subsequent sections.

5.3.1 The DIHANA corpus

The Spanish Corpus DIHANA [Benedı et al., 2006] is composed of 900 dialogues about railway information such as trip timetables and prices for different destinations in Spain. The scenarios where obtained from 225 users (153 males and 72 females). The acquisition was performed using the Wizard of Oz strategy [Fraser & Gilbert, 1991]. In this strategy, the user believes that he is interacting with the real system while a human agent is simulating the behaviour of the system.

The acquisition process had 5.3 hours of speech signal which resulted in 6,278 user turns and 9,129 system turns, with an average of 7 user turns and 10 system turns per dialogue. The vocabulary size is 839 words with 9 words per user turn and 20 for

The DIHANA Corpus	
Dialogues	900
Speakers	225
User turns	6,278
System turns	9,129
Words	48,243
Vocabulary size	839

Table 5.4: Statistics from the DIHANA corpus

system turns. The main characteristics of the corpus are shown in Table 5.4.

As we mentioned in Section 5.1.2, this corpus is based on the three level Interchange Format (IF) defined in the C-STAR project. In the DIHANA project, the first layer represents the intention of the segment, which corresponds to a generic description of the dialogue act. More precise semantic information are described in the second and third layer. The second level contains the repository of information implicit in the segment, whereas the third level store the specific data that appear in the segment Alcácer et al. [2005]. Table 5.5 shows a sample of an annotated dialogue (translated into English) from the corpus.

Using this three level labelling scheme and making the distinction between user and system labels, the annotation task resulted in 248 different dialogue act labels (153 for user turns and 95 for system turns). Since the third level contains very detailed information, an alternative labelling scheme using only the first two levels is also considered in experiments [Tamarit et al., 2011]. This two level scheme, which we have adopted for our experiments, contains 72 different labels (45 for user and 27 for system). Also, by considering only the first level, which contains general information regarding the dialogue act, there is a total of 16 labels (7 for user and 9 for system).

The corpus was also preprocessed to reduce complexity. In this case, all words were transformed to lower case, punctuation marks where separated from words, and a process of name-entity recognition was applied so as to categorise entities such as town names, dates,and hours.

Segment	Translated Utterances		
	Level 1	Level 2	Level 3
S1	Welcome to the railway information system. How may I help you?		
	Opening	*Nil*	*Nil*
U1	I want to know the departure hours from Valencia		
	Question	*Departure-Hour*	*Origin*
U2	to Madrid		
	Question	*Departure-Hour*	*Destination*
U3	arriving on May 15th, 2004.		
	Question	*Departure-Hour*	*Day*
S2	Do you want to leave on Saturday, May 15th 2004?		
	Confirmation	*Day*	*Day*
U4	Yes.		
	Acceptance	*Day*	*Nil*
S3	Consulting times for trains from Valencia to Madrid on Saturday, May 15th 2004.		
	Confirmation	*Departure-Hour*	*Destination, Day, Origin*
S4	Wait a moment please.		
	Waiting	*Nil*	*Nil*
S5	There are several trains. The first one leaves at 7:45 and arrives at 11:14, and the last one leaves at 18:45 and arrives at 22:18.		
	Answer	*Departure-Hour*	*Arrival-Hour*, *Departure-Hour*, *Order-Number*, *Number-Trains*
S6	Do you need anything else?		
	Consult	*Nil*	*Nil*
U5	Yes, I want to know the fare for the train leaving at 7:45.		
	Question	*Fare*	*Departure-Hour*
S7	The train in the tourist class costs 35.50 euros.		
	Answer	*Fare*	*Class, Fare*
S8	Do you need anything else?		
	Consult	*Nil*	*Nil*
U6	No, thank you.		
	Closing	*Nil*	*Nil*
S9	Thank you for using this service. Have a nice day.		
	Closing	*Nil*	*Nil*

Table 5.5: A sample dialogue from the DIHANA corpus in English.

5.3.2 Methodology

Our goal in the following experiments is to apply the Dirichlet process mixture model to the task of automatic classification of dialogue acts in the DIHANA corpus. The two level scheme of DA labels was chosen for analysis of results, so there were 27 system DA's and 45 user dialogue acts. We consider utterances as bags of words (see Section 5.2.3), so syntactic features are not considered here. Classification is based on word counts in each utterance, therefore utterances with similar words will in general appear in the same cluster.

According to what we discussed in the previous sections, a Dirichlet process mixture model can be described as an infinite dimensional counterpart of a finite mixture model. As with all nonparametric models, in the DPMM we can have a potentially infinite number of clusters. Since we are working with word histograms, a convenient choice for the words appearing in each cluster would be the Dirichlet-Multinomial distribution explained in Section 4.1.1. In other words, the appearance of words in utterances that belong to the same cluster is governed by a Multinomial distribution with a Dirichlet prior. A small example of utterance classification using (finite) mixture models was presented in Section 3.3.1.

When using the DPMM, there are two hyperparameters which greatly affect the number of clusters that are obtained at the end. The first one is the γ parameter of the Dirichlet process. As we saw in Section 3.2.5, the expected number of clusters is directly dependent on the value of this parameter. So small changes in γ will affect the number of clusters, which will in turn influence the error rate of the whole approach. However, as we will demonstrate later, by using the Gibbs sampling algorithm described in Algorithm 2, the initial value chosen for γ had little effect on the converged value of this hyperparameter.

The second important hyperparameter is the parameter vector of the Dirichlet prior in the Dirichlet-Multinomial distribution. As we explained in Section 2.3.3, this parameter controls how concentrated the distribution is around its expected value. Choosing 1 as its value results in all sets of probabilities in that simplex being equally likely (the distribution over any set of probabilities is uniform). On the other hand, choosing a large value for the parameter will end up in uniform distributions having higher probabilities. For small values, one would more probably wind up with distributions concentrated on one of their components (highly peaked at corners of the simplex). Therefore, the value chosen for this parameter will affect how utterances are clustered together, and thus how many clusters we obtain overall.

Whereas the initial value for γ parameter of the Dirichlet process seemed to have little effect on the final results, different values of the Dirichlet-Multinomial parameter resulted in different number of clusters. So we divided the corpus into training and test sets. Since the corpus was large, half of the corpus was set aside for training and the rest was used for validation purposes. During the training phase of the project, we trained our classifier using various values for the concentration parameter of the Dirichlet-Multinomial distribution. The parameter with the lowest classification error was chosen for the validation phase of the project.

We used Algorithm 4 to draw samples from the DPMM and to estimate values of the γ hyperparameter of the model. the algorithm starts by randomly assigning utterances to an initial number of clusters. As we will demonstrate later, this initial value does not affect our final results. At each step, a single utterance is removed from the cluster it has been assigned to, and is re-assigned to one of the clusters based on the words appearing in each cluster. Since we are using a Dirichlet-Multinomial distribution for word occurrences in clusters, we should apply the corresponding predictive distribution of Eq. (4.13) as the posterior f_p in the algorithm. This process was repeated until convergence, where the convergence criterion was met when utterances moving between clusters did not change the total number of components for 5 consecutive iterations.

After every 15 scans of the Gibbs sampler, we updated the Dirichlet-Multinomial parameters of each cluster so as they would further resemble the data points they contain. This was done using our Newton-Raphson technique with an exponential mapping described in Section 4.1.2. So by using the Newton-Raphson steps defined in Eq. (4.45), we found values of the hyperparameter which made the observations in that cluster the most likely possible.

At the end of each iteration, we also had to update the γ parameter of the DP for the next run of the sampler. Based on the final step of 4, we ran 20 different trials of the sampler, and at the end of each run, recorded the number of clusters found. Solving Eq. (4.72) by averaging over these values results in a new value of γ which we used for the next iteration of the Gibbs sampler.

The whole iteration scheme presented above was repeated 100 times, even though convergence was reached much earlier. We used two programming languages for coding our experiments. The first part, which involved reading text data and extracting information from the dialogues, was done using PYTHON due to its simplicity in file I/O and easy to use regular expressions. These data where then imported into MATLAB where all the algorithms for working on the utterances were implemented in this environment.

Before going on to discuss our results, it should be mentioned that a similar work using the Dirichlet process has been already applied to the DIHANA corpus by Crook et al. [2009]; however, in their approach it is not specified what hyperparameters they have chosen for the model, and how they have obtained these values. As mentioned previously, the final results are highly sensitive to the initial values chosen. Therefore, it is not clear how 'unsupervised' their approach can be unless they explain the model in more detail. Moreover, they apply their algorithm on all dialogues together, instead of separating user and system utterances. As we will show below, classification of system utterances normally yields higher accuracy than user utterances because they are generated from a well defined pattern. So by mixing user and system utterances, the results may not actually tell us how well the algorithm performs on user utterances, which is more of a challenge in DA classification approaches.

In what follows, we will discuss the main results that we obtained during our experiments. Apart from the two error criteria that we discussed in Section 5.2 (the percentage of correctly classified utterances, and the confusion matrix), following Crook et al. [2009] two other evaluation criteria were also introduced. The *intra-cluster similarity* measures the similarity between utterances that belong to the same cluster. This is done by averaging over the Euclidean distance between every pair of utterances in cluster k, or in formal terms:

$$S_k^{'} = \frac{1}{2n_k} \sum |u_i - u_j| \tag{5.5}$$

where n_k represents the number of utterances in cluster k, and u_i is the vectorial representation of the i'th utterance, signifying the number of times each word has appeared in that utterance. Not only should we seek to make utterances in each cluster more similar to each other, but it is also important to increase the *inter-cluster similarity* so that clusters are as far apart as possible. This metric is calculated by averaging over the Euclidean distance between the centres of every pair of clusters:

$$S_k^{''} = \frac{1}{2K} \sum |C_i - C_j| \tag{5.6}$$

where C_i and C_j are centres of two typical clusters and K is the total number of clusters found by the algorithm. These criteria are evaluated for both training and validation phases as we will describe below.

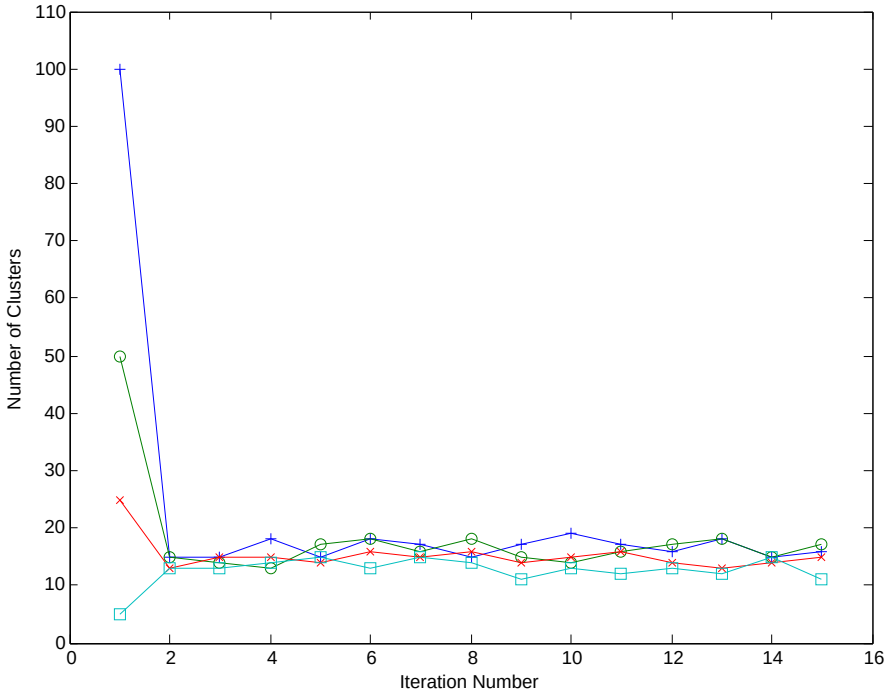

Figure 5.1: Initialization of the number of clusters. As demonstrated here, this value had no significant effect on the final converged value.

5.3.3 Results of Training

As mentioned before, there are three parameters that we should initialize for our algorithm to work. These were the number of clusters to start off with, the Dirichlet process parameter and the Dirichlet-Multinomial parameters of each cluster. As for the last one, we ran the algorithm for 200 different values of the cluster parameter between 0 and 10, and tried to find the parameter with the lowest error rate, highest inter-cluster similarity and lowest intra-cluster similarity overall. With these criteria in mind, the best parameters were found to be 34/190 or 0.179 for system utterances and 40/705 or 0.057 for user utterances.

Initial Number of Clusters

The initial number of clusters which we fed into our algorithms seemed to have no significant effect on the final converged value. We started with 5, 25, 50 and 100 clusters holding all other parameters fixed and, as can be seen in Figure 5.1, they all converged to the same number of clusters after very few steps.

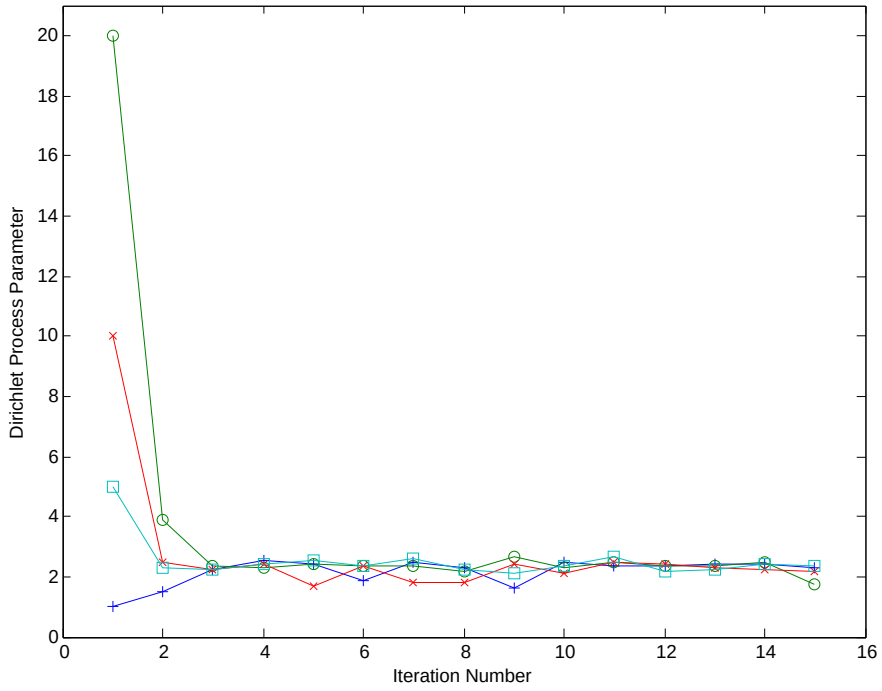

Figure 5.2: Initialization of the Dirichlet process parameter γ. As demonstrated here, this value had no significant effect on the final converged value either.

Initial Value of the DP Parameter

The second parameter which we had to initialize was the Dirichlet process parameter. Like the previous case, no significant impact was observed when given different starting values. Figure 5.2 shows results of 4 executions starting off with $\gamma = 1$, 5, 10 and 20. In all cases, the algorithm converged to the same γ value after a relatively few number of iterations.

5.3.4 Final Results

With the initial values found in the training phase, we then started the algorithm on the test set and iterated over the Gibbs sampler for 100 iterations. Although some fluctuations can be observed among the values obtained at different iterations, but they all seem to be in a specific range, which means that the algorithm has a high rate of convergence. At each step, we recorded the value of the Dirichlet process parameter, the error rates, similarity rates and the number of clusters found. The results along with their corresponding graphs are discussed below.

Dirichlet Process Parameter

We started the DP parameter γ with 10, even though we could have initialized it with any other number due to our discussion in Section 5.3.3. The values obtained were all very close, and mostly between 2 and 4. However, by observing Figures 5.3 and 5.4, it can be seen that after around 65 iterations over user utterances and 70 iterations over system utterances, the value of γ starts to fluctuate less than the initial steps. So by averaging over γ after this burn in period, we find $\gamma = 9.058$ for the user set and $\gamma = 2.675$ for the system set.

Number of Clusters

As we discussed in Section 3.2.5, the γ parameter of the DP directly affects the number of clusters obtained. Once again, the initial value assigned to this parameter had been shown to have no effect on the final converged value. Figures 5.5 and 5.6 show how the number of clusters change over time for system and user utterances. It can be observed that with γ starting to converge, the variation in the number of clusters gets lower. So the converged number of clusters was 25.48% for system utterances and 49.86% for user utterances.

By only considering the number of clusters, the results obtained here are very close to the actual numbers. In fact, based on the two level annotation scheme of the DIHANA corpus, there were 27 and 45 dialogue acts for system and user utterances, respectively. However, the number of clusters is not the only factor which demonstrates the performance of the algorithm. The next set of results show different error criteria evaluated for each of the obtained clusters.

Performance Analysis

After finding the different number of clusters and assigning utterances to each one of them, we labelled each cluster with the DA tag which had the largest proportion of utterances within it. This is important for the next steps where we want to evaluate the performance of the approach.

Performance analysis was based on four different evaluation criteria described at the beginning of this section. The first one was the calculation of the error rate based on the confusion matrix as formulated in Eq. (5.1). In our context, the false negatives for

the i'th cluster were all utterances that should have been in that cluster, but ended up in some other cluster with another DA label. On the other hand, the false positives were all instances that we assigned to the i'th cluster while they actually had a different DA label than the one assigned to that cluster. Figures 5.7 and 5.8 show the mean error rate of the clusters obtained in each iteration of the Gibbs sampler. Accordingly, the error rate after convergence had an average of 0.013 for system utterances and 0.015 for user utterances.

A more intuitive criterion for analysing results is the percentage of correctly classified utterances in each cluster. Figures 5.9 and 5.9 demonstrate how well our approach performed when clustering user and system utterances. As with the previous case, for each iteration we plot the average of classification accuracy in all clusters. After the burn in period, the average classification precision turned out to be 83.85% for system utterances and 62.89% for user utterances.

Last but not least, intra-cluster and inter-cluster similarity measures as explained in Eqs. (5.5) and (5.6) were calculated. For system and user utterances, the average intra-cluster similarity was 114.833 and 102.365 and the average inter-cluster similarity was 68.556 and 70.547 respectively. Remember that a good classification algorithm is one which besides low error rates, has low intra-cluster similarity (utterances in each cluster are closer to each other) and high inter-cluster similarity (clusters are as far apart as possible). Figures 5.11-5.14 show the results of our similarity analysis for both type of utterances.

It thus follows that the Dirichlet process mixture model was able to classify dialogue acts in the DIHANA corpus with acceptable precision. The main reason that the algorithm performed so well in classifying system utterances, compared to user utterances, is that in the latter case utterances come from different sources, whereas system utterances are mostly predefined and have many similar structures within them. Also, updating the cluster parameters after a predetermined number of steps makes them more specialized, in the sense that they will further reflect the properties of utterances they hold. This results in utterances wandering less among clusters, an increase in inter-cluster similarity and decrease in the intra-cluster similarity measures.

5.4 Conclusion

In this section, we discussed the problem of dialogue act classification in spoken dialogue systems. We reviewed some of the most important approaches used for automatic DA classification, along with results of applying them to different dialogue corpora. We next used the concepts presented in the previous chapters to analyse the performance of nonparametric methods in this area. We tried to model dialogue utterances with Dirichlet process mixture models by treating utterances as bag of words.

Our results show that the DPMM can have acceptable results when applied to dialogue act classification. The model was able to correctly classify 63% of the user utterances and 84% of system utterances. This difference in performance is quite understandable, because classifying user utterances is more of a challenge because they come from different sources each with different language competencies. However, with the appropriate hyperparameters, the DPMM was almost capable of finding the correct number of two-level DA classes necessary. Even though the results sound promising, the DIHANA corpus which we used for our experiments is restricted to a specific task, it is still unclear whether or not the method could be applied to more general corpora.

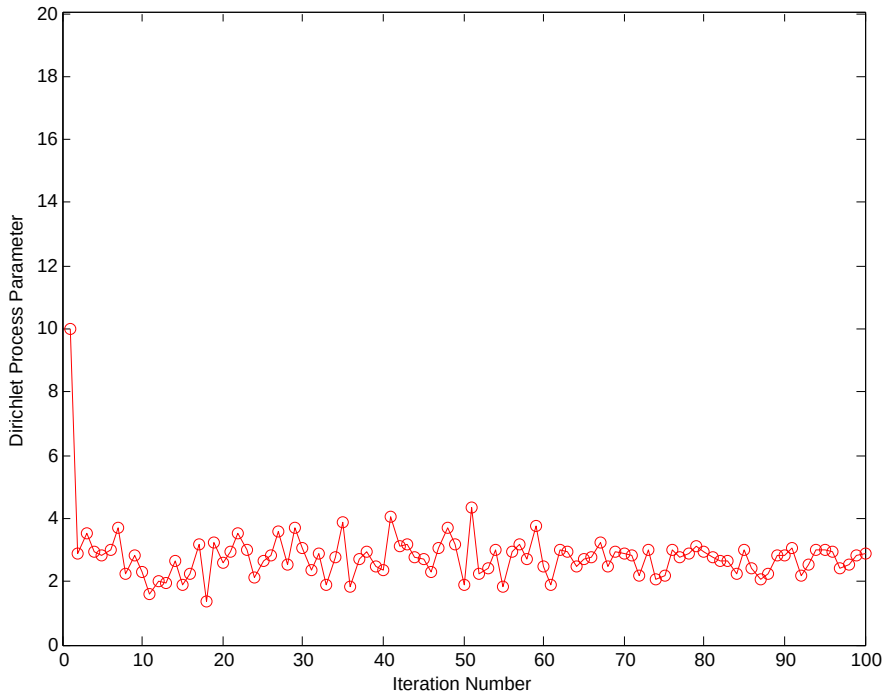

Figure 5.3: The value of the γ parameter of the Dirichlet process in different iterations of the Gibbs sampler over system utterances.

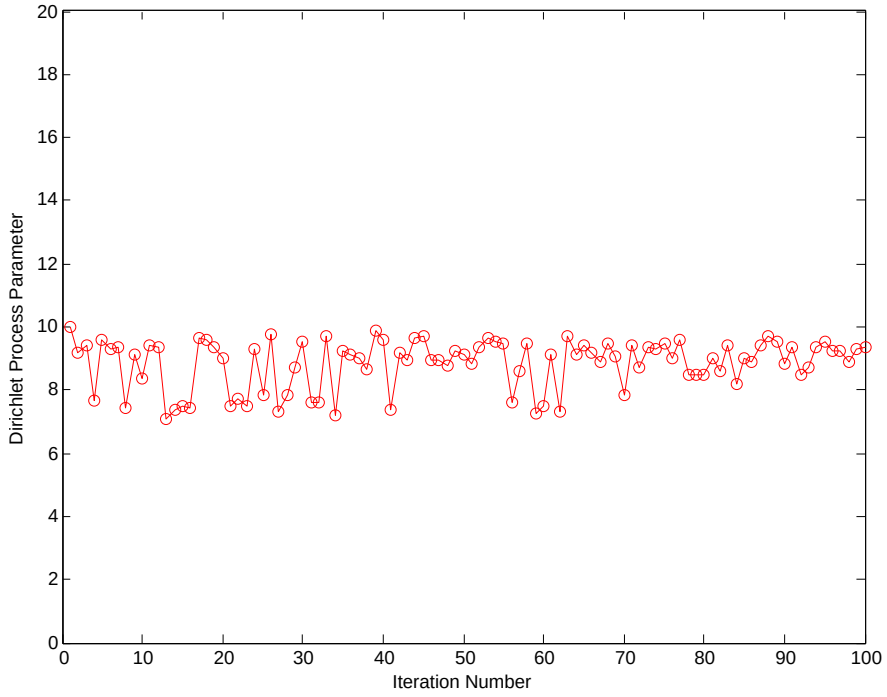

Figure 5.4: The value of the γ parameter of the Dirichlet process in different iterations of the Gibbs sampler over user utterances.

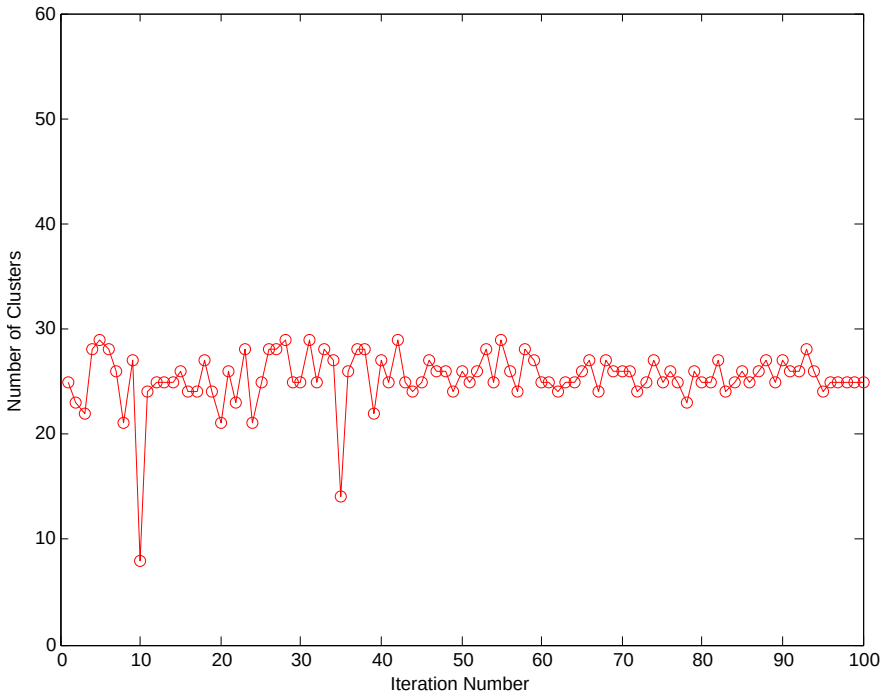

Figure 5.5: Number of clusters obtained in each iteration of the Gibbs sampler over system utterances.

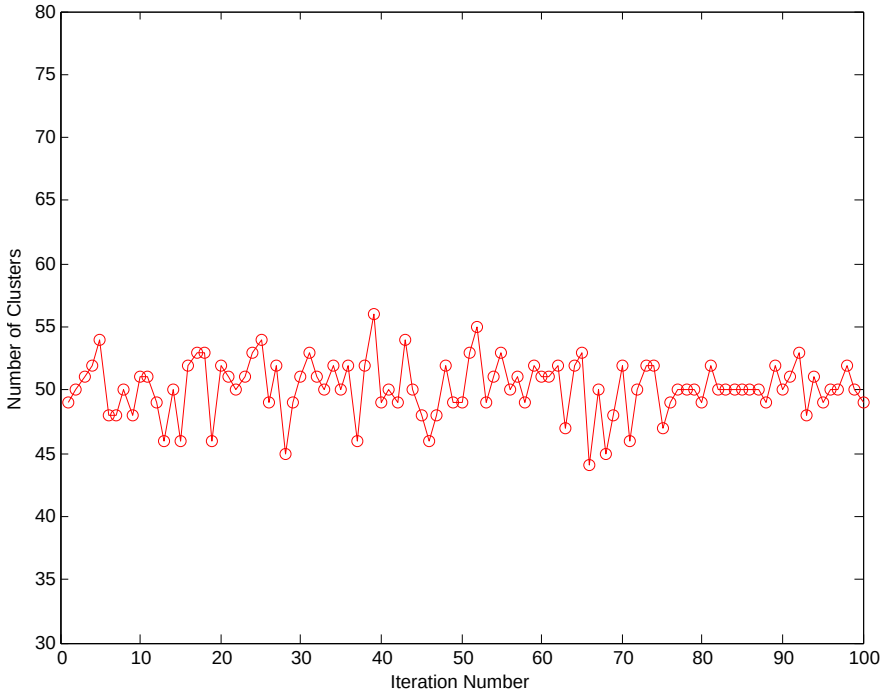

Figure 5.6: Number of clusters obtained in each iteration of the Gibbs sampler over user utterances.

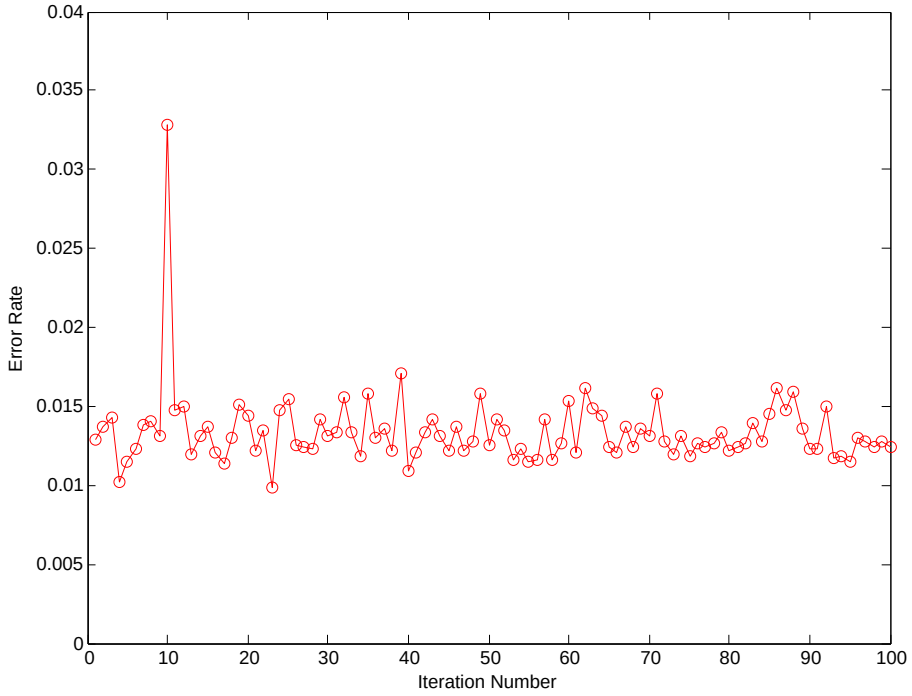

Figure 5.7: Classification error based on the confusion matrix for system utterances.

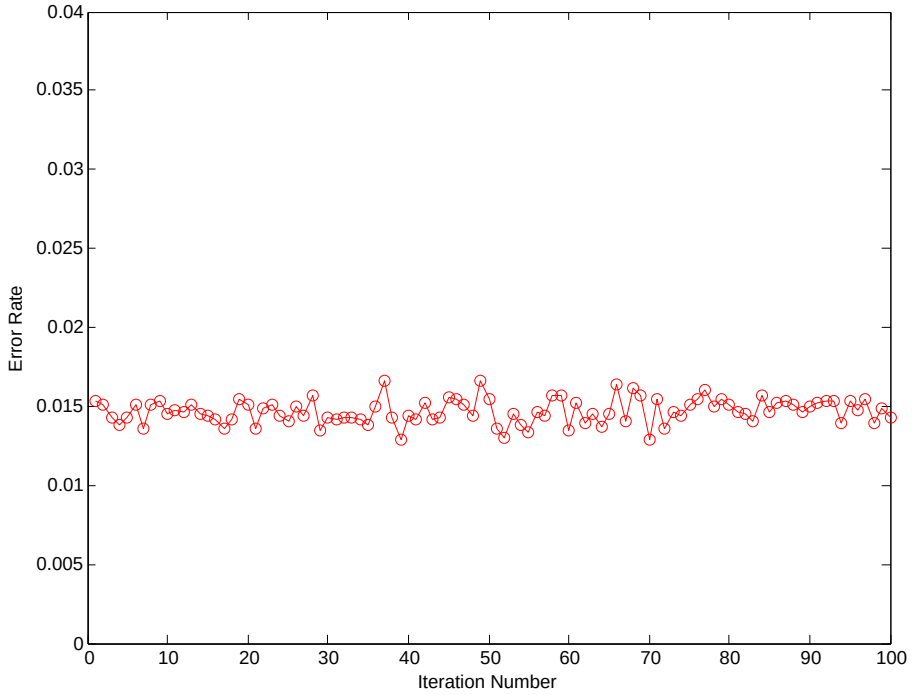

Figure 5.8: Classification error based on the confusion matrix for user utterances.

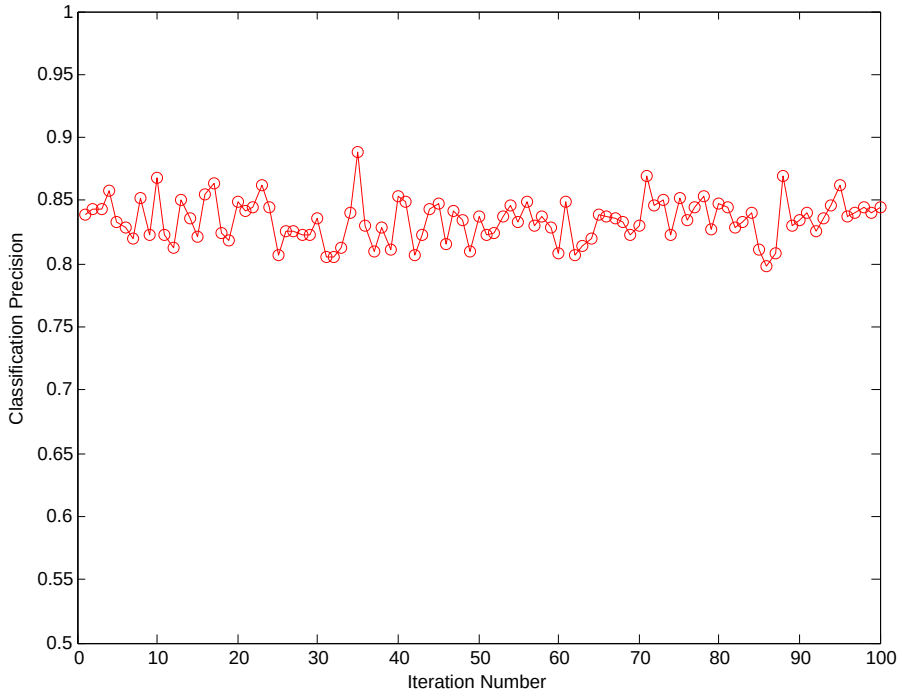

Figure 5.9: Classification accuracy for system utterances in each iteration of the Gibbs sampler.

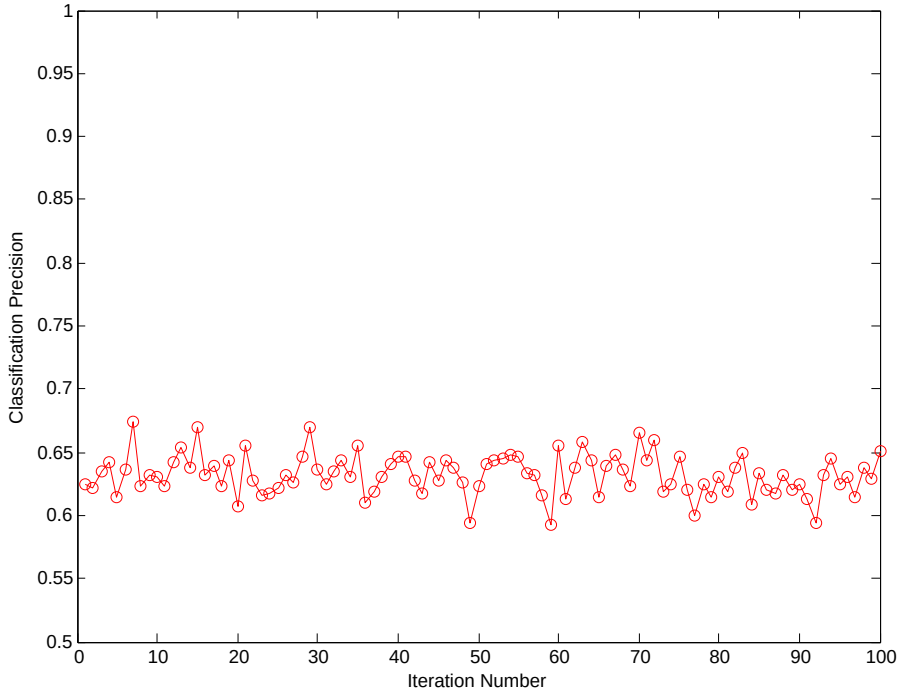

Figure 5.10: Classification accuracy for user utterances in each iteration of the Gibbs sampler.

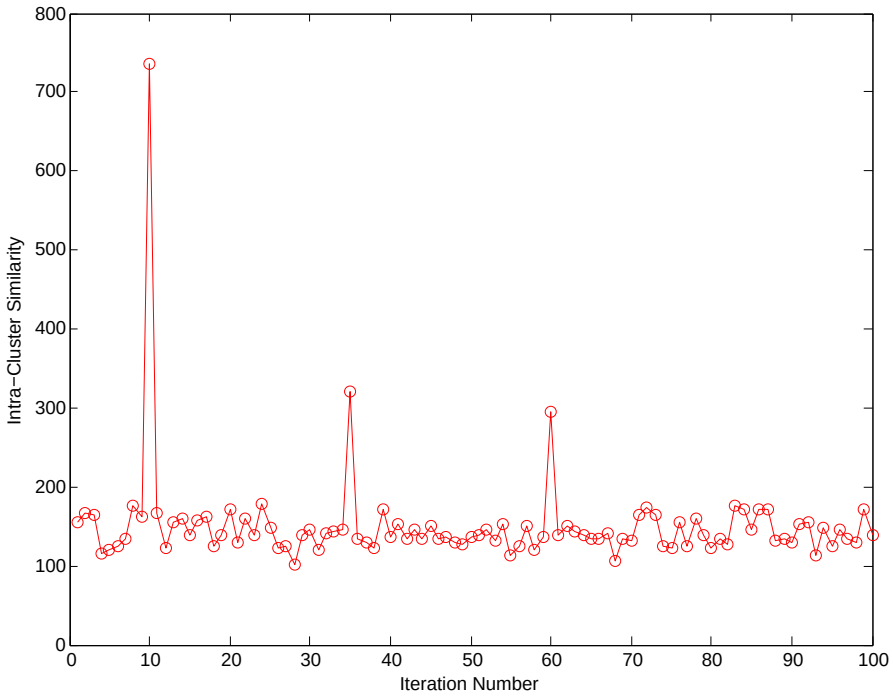

Figure 5.11: Intra-cluster similarity of clusters of system utterances during the evolution of the Gibbs sampler.

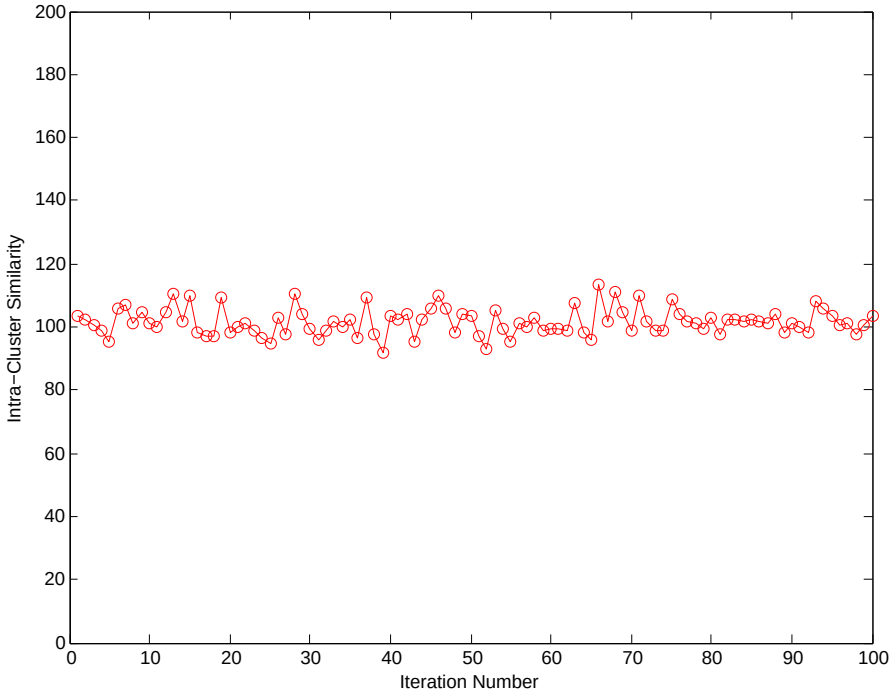

Figure 5.12: Intra-cluster similarity of clusters of user utterances during the evolution of the Gibbs sampler.

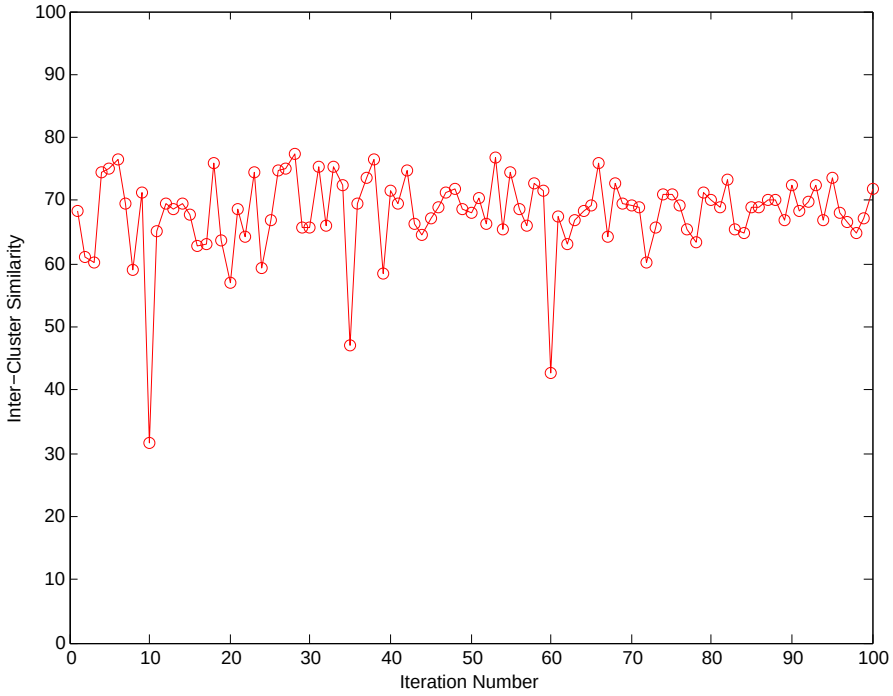

Figure 5.13: Inter-cluster similarity between clusters of system utterances at each step of the Gibbs sampler.

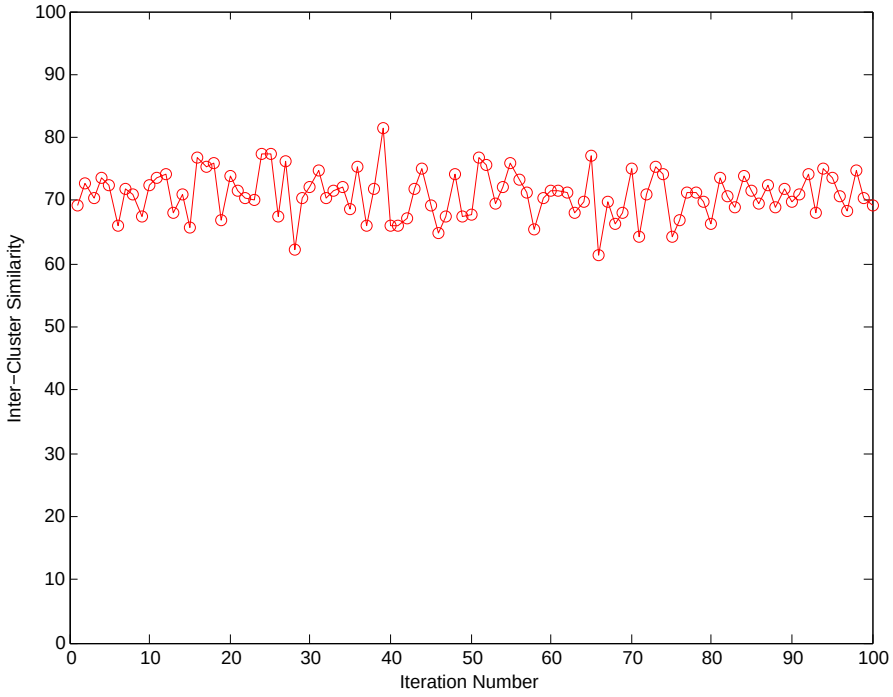

Figure 5.14: Inter-cluster similarity between clusters of user utterances at each step of the Gibbs sampler.

Chapter 6

Conclusion and Future Work

This thesis considers the application on nonparametric Bayesian methods to the problem of automatic dialogue act classification. We discussed the Dirichlet process mixture model, along with methods for sampling and estimating its key hyperparameters in Chapters 3 and 4. In this final chapter, we will discuss the main contributions and results of this work, along with directions for future research.

Automatic dialogue act classification with high precision is still an open problem in spoken dialogue systems. It has turned into a classic challenge in classification, where the goal is to extract important features from a set of manually annotated dialogues, in order to use them for predicting the presence of particular DA's in future utterances. As we discussed in Chapter 5, several approaches have been already taken for correctly predicting the DA's in various corpora with different annotation schemes. However, so far no approach has been ideal or fully robust.

Conceptually, the contributions of this thesis can be broken into two main parts. The first part, which consists of Chapter 4, discussed two new approaches for estimating hyperparamters in the Dirichlet-Multinomial distribution. This distribution is the building block of all models that deal with observations drawn from a Multinomial distribution (such as word counts in each DA component), where the Multinomial proportions are thought to have been drawn from its conjugate prior, the Dirichlet distribution. Therefore, not only can we use them for finite mixture models, but we can also apply them to their infinite dimensional counterpart, the Dirichlet process mixture model.

Hyperparameter estimation in the Dirichlet-Multinomial distribution is not a new problem in statistical computations; in fact, there are already several methods available for this task. Nevertheless, as noted by previous authors, they all seem to have over-

looked one important implementation constraint; that the parameters in a Dirichlet distribution must always remain positive. The two new methods that we introduced overcome this limitation by enforcing the constraints into the Newton-Raphson optimization technique. In the first approach, we used an exponential mapping to map each parameter into an exponential parameter where the domain is the set of all real numbers, but the output range is always positive. So the problem is transformed into a regular optimization problem with no constraints.

In the second method, we tackled the problem from another angle by using a logarithmic barrier function which forces the maximization problem to always return positive values. Simplified Newton-Raphson steps were also provided for approximate optimization using this technique. However, both of the methods need to be implemented and compared with other existing algorithms in terms of performance and convergence speed.

The second set of contributions are the application of the Dirichlet process mixture model to the problem of automatic DA classification in the DIHANA corpus. In this model, the number of clusters is governed by the Dirichlet process, which allows for a (potentially) infinite number of clusters. Within each cluster, the utterances are assumed to have a Dirichlet-Multinomial distribution, so utterances with similar word patterns will normally fall into the same DA cluster. The parameters of all components where the same at first, but after a few iterations, were updated to further resemble the utterances present in them. This updating was done using our exponential mapping estimation technique. The results were quite remarkable; the number of clusters obtained was very close to the true number, with 25 clusters for system utterances and 50 clusters for user utterances, whereas the real values were 27 and 45 respectively.

The classification precision was also quite acceptable. The algorithm was capable of correctly classifying 63% of user utterances and 84% of system utterances. This shows that even without considering syntactic features of utterances, word counts alone can largely contribute to the classification of dialogue acts using the Dirichlet process mixture models.

However, this is just a first step in the application of nonparametric Bayesian methods to the challenging problem of DA classification. For the hyperparameter estimation techniques presented in this work, it is necessary to implement them and compare them to other estimation techniques used in the literature as to see how they compare in terms of speed and accuracy. Also, further research needs to be done regarding the t variable in the logarithmic barrier approach. As we mentioned in Section 4.1.2, small values of this variable will result in poor approximations, whereas larger values increase

the difficulty when optimizing using the Newton-Raphson method.

Our experiments considered utterances as simple bags of words, neglecting all syntactic features a spoken utterance may have. One can alternatively use other features to increase classification accuracy. N-grams of words can further help classify utterances into relative DA clusters. These n-grams can be obtained from available sources or even from the corpus itself by considering different sequences of words in that corpus. This second approach has the benefit of generating n-grams related to the task we are dealing with in whatever language the corpus is compiled. DA n-grams can also be extracted from the annotated corpus, so there would be a more logical connection between the DA we are trying to predict and the previous DA's in the progression of the conversation.

Last but not least, our experiments were restricted to a task oriented corpus with dialogues concentrating on train service inquiries. Further experiments need to be done in order to evaluate the performance of the DPMM on corpora with larger vocabularies and more general topics like the Switchboard corpus. one might even wonder how well could this model perform on new communication mediums such as twitter conversations. The collection of 1.3 million twitter conversations recently made available by Microsoft Research, provides a new challenge for dialogue act labelling in a completely unsupervised open domain.

Bibliography

Abramowitz, M., & Stegun, I. A. (1972). *Handbook of Mathematical Functions*, vol. 55 of *Applied Mathematics Series*. Dover.

Aitchison, J., & Dunsmore, I. R. (1975). *Statistical prediction analysis*. Cambridge University Press.

Alcácer, N., Benedı, J., Blat, F., Granell, R., Martınez, C., & Torres, F. (2005). Acquisition and labelling of a spontaneous speech dialogue corpus. In *Proceedings of SPECOM*, (pp. 583–586).

Aldous, D. J. (1985). Exchangeability and related topics. *Lecture Notes in Mathematics*, *1117*, 1–198.

Allen, J. (1987). *Natural language understanding*. Benjamin/Cummings Publishing Company, Inc.

Alpaydin, E. (2004). *Introduction to machine learning*. The MIT Press.

Andrieu, C., De Freitas, N., Doucet, A., & Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine Learning*, *50*(1), 5–43.

Ang, J., Dhillon, R., Krupski, A., Shriberg, E., & Stolcke, A. (2002). Prosody-based automatic detection of annoyance and frustration in human-computer dialog. In *Seventh International Conference on Spoken Language Processing*.

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

Athreya, K. B., & Lahiri, S. N. (2006). *Measure Theory and Probability Theory*. Springer Texts in Statistics.

Austin, J. L. (1962). *How to do things with words*. Harvard University Press.

Balakrishnan, N., & Nevzorov, V. B. (2003). *A Primer on Statistical Distributions*. Wiley-Interscience.

Banuelos, R. (2003). *Measure Theory and Probability*. Lecture notes, Department of Mathematics, Purdue University.

Barankin, E., & Maitra, A. (1963). Generalization of the Fisher-Darmois-Koopman-Pitman theorem on sufficient statistics. *Sankhyā: The Indian Journal of Statistics, Series A*, *25*(3), 217–244.

Beal, M. J. (2003). *Variational algorithms for approximate Bayesian inference*. Ph.D. thesis, University College London.

Benedı, J., Lleida, E., Varona, A., Castro, M., Galiano, I., Justo, R., López, I., & Miguel, A. (2006). Design and acquisition of a telephone spontaneous speech dialogue corpus in Spanish: DIHANA. In *Fifth International Conference on Language Resources and Evaluation (LREC)*, (pp. 1636–1639).

Bickerton, D. (1992). *Language & species*. University of Chicago Press.

Billingsley, P. (1995). *Probability and Measure*, vol. 35 of *Wiley series in probability and statistics*. Wiley.

Bishop, C. (2006). *Pattern recognition and machine learning*, vol. 4. Springer New York:.

Blackwell, D., & MacQueen, J. B. (1973). Ferguson Distributions Via Polya Urn Schemes. *Annals of Statistics*, *1*(2), 353–355.

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

Blei, D., Griffiths, T., Jordan, M., & Tenenbaum, J. (2004). Hierarchical topic models and the nested Chinese restaurant process. *Advances in neural information processing systems*, *16*, 106.

Blei, D. M., & Frazier, P. I. (2009). Distance Dependent Chinese Restaurant Processes. *Arxiv preprint arXiv:0910.1022*.

Blei, D. M., & Jordan, M. I. (2006). Variational Inference for Dirichlet Process Mixtures. *Bayesian Analysis*, *1*(1), 121–144.

Bohmer, P. (1939). *Differenzengleichung und bestimmte Integrale*. KF Koehler.

Bolstad, W. M. (2010). *Understanding Computational Bayesian Statistics*. Wiley.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Brown, L. D. (1986). *Fundamentals of statistical exponential families: with applications in statistical decision theory*. Lecture Notes–Monograph Series. Institute of Mathematical Statistics.

Bunt, H. (1994). Context and dialogue control. *Think Quarterly*, *3*(1), 19–31.

Burden, R., & Faires, J. (2010). *Numerical Analysis*. Brooks Cole, 10 ed.

Casella, G., & George, E. I. (1992). Explaining the Gibbs Sampler. *American Statistician*, *46*(3), 167–174.

Chen, M., Shao, Q., & Ibrahim, J. G. (2000). *Monte Carlo Methods in Bayesian Computation*, vol. 95 of *Springer Series in Statistics*. Springer-Verlag.

Chomsky, N. A. (2002). The evolution of language. In *the 4th International Conference on the Evolution of Language, Harvard University*.

Clark, D. R., & Thayer, C. A. (2004). A primer on the exponential family of distributions. *Casualty Actuarial Society Spring Forum*, (pp. 117–148).

Core, M., & Allen, J. (1997). Coding dialogs with the DAMSL annotation scheme. In *AAAI Fall Symposium on Communicative Action in Humans and Machines*, (pp. 28–35).

Cox, R. T. (1946). Probability, Frequency and Reasonable Expectation. *American Journal of Physics*, *14*, 1–13.

Craggs, R., & Wood, M. (2003). Annotating emotion in dialogue. In *Proceedings of the Fourth SIGdial Workshop on Discourse and Dialogue*, (pp. 218–225).

Crook, N., Granell, R., & Pulman, S. (2009). Unsupervised classification of dialogue acts using a dirichlet process mixture model. *Proceedings of the SIGDIAL 2009 Conference on The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue - SIGDIAL '09*, (September), 341–348.

Daelemans, W., Zavrel, J., van der Sloot, K., & van Den Bosch, A. (2007). Timbl: Tilburg memory-based learner. *version*, *6*, 07–03.

Darmois, G. (1935). Sur les lois de probabilité à estimation exhaustive. *C.R. Acad. Sci. Paris*, *260*, 1265–1266.

Dawkins, R. (1982). *The extended phenotype: the gene as the unit of selection*. Freeman.

De Finetti, B. (1931). *Funzione Caratteristica Di un Fenomeno Aleatorio*, vol. 4 of *6. Memorie*, (pp. 251–299). Academia Nazionale del Linceo.

Dorazio, R. (2009). On selecting a prior for the precision parameter of Dirichlet process mixture models. *Journal of Statistical Planning and Inference*, *139*(9), 3384–3390.

Dorazio, R., Mukherjee, B., Zhang, L., Ghosh, M., Jelks, H., & Jordan, F. (2008). Modeling unobserved sources of heterogeneity in animal abundance using a Dirichlet process prior. *Biometrics*, *64*(2), 635–644.

Doshi, F. (2009). *The Indian Buffet Process: Scalable Inference and Extensions*. Ph.D. thesis, Cambridge.

Doss, H. (2008). *Estimation of Bayes factors for nonparametric Bayes problems via Radon-Nikodym derivatives*. Technical report, University of Florida.

Dunbar, R. (1998). *Grooming, gossip, and the evolution of language*. Harvard Univ Pr.

Durbin, R. (1998). *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge University Press.

Elkan, C. (2006). Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution. In *Proceedings of the 23rd international conference on Machine learning*, (pp. 289–296).

Escobar, M. (1994). Estimating normal means with a Dirichlet process prior. *Journal of the American Statistical Association*, *89*(425), 268–277.

Escobar, M., & West, M. (1995). Bayesian Density Estimation and Inference Using Mixtures. *Journal of the american statistical association*, *90*(430).

Everitt, B. S., & Hand, D. J. (1981). *Finite Mixture Distributions*, vol. 2. Chapman and Hall.

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

Fernández, R., Ginzburg, J., & Lappin, S. (2005). Using machine learning for non-sentential utterance classification. In *6th SIGdial Workshop on Discourse and Dialogue*.

Fishel, M. (2006). Dialogue Act Recognition Techniques. *GSLT/NGSLT course on dialogue systems: Linkoping University, Sweden*.

Fisher, R. (1922). On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, *222*(594-604), 309.

Fox, E. B. (2009). *Bayesian Nonparametric Learning of Complex Dynamical Phenomena*. Ph.D. thesis, Massachusetts Institute of Technology.

Franks, J. M. (2009). *A (terse) introduction to Lebesgue integration*. American Mathematical Society.

Fraser, N., & Gilbert, G. (1991). Simulating speech systems* 1. *Computer Speech & Language*, *5*(1), 81–99.

Fukada, T., Koll, D., Waibel, A., & Tanigaki, K. (1998). Probabilistic dialogue act extraction for concept based multilingual translation systems.

Gelfand, A., Kottas, A., & MacEachern, S. (2005). Bayesian nonparametric spatial modeling with Dirichlet process mixing. *Journal of the American Statistical Association*, *100*(471), 1021–1035.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (2004). *Bayesian Data Analysis*. Chapman and Hall, 2nd ed.

Ghahramani, Z., Griffiths, T. L., & Sollich, P. (2007). *Bayesian nonparametric latent feature models*, vol. 8, (pp. 1–19). Oxford University Press.

Ghosh, J. K., & Ramamoorthi, R. V. (2003). *Bayesian Nonparametrics*. Springer Series in Statistics. New York: Springer-Verlag.

Goldwater, S., Griffiths, T. L., & Johnson, M. (2006). Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, vol. 18, (pp. 459–466). MIT Press.

Grau, S., Sanchis, E., Castro, M., & Vilar, D. (2004). Dialogue act classification using a Bayesian approach. In *Proceedings of the 9th International Conference Speech and Computer*, (pp. 495–499).

Griffiths, T., & Ghahramani, Z. (2005). Infinite Latent Feature Models and the Indian Buffet Process. *Advances in Neural Information Processing Systems*, *18*(17), 475.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, *57*(1), 97–109.

Hazewinkel, M. (Ed.) (2002). *Encyclopaedia of Mathematics*. Springer-Verlag Berlin Heidelberg New York.

Hjort, N. L., Holmes, C., Muller, P., & Walker, S. G. (Eds.) (2010). *Bayesian Nonparametrics (Cambridge Series in Statistical and Probabilistic Mathematics)*. Cambridge University Press.

Huang, J. (2005). Maximum Likelihood Estimation of Dirichlet Distribution Parameters.

Ishwaran, H., & James, L. (2001). Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, *96*(453), 161–173.

Ishwaran, H., & Zarepour, M. (2000a). Markov chain Monte Carlo in approximate Dirichlet and beta two-parameter process hierarchical models. *Biometrika*, *87*(2), 371–390.

Ishwaran, H., & Zarepour, M. (2000b). Markov chain Monte Carlo in approximate Dirichlet and beta two-parameter process hierarchical models. *Biometrika*, *87*(2), 371.

Ishwaran, H., & Zarepour, M. (2002). Exact and approximate sum representations for the Dirichlet process. *The Canadian Journal of Statistics/La Revue Canadienne de Statistique*, *30*(2), 269–283.

Ivanovic, E. (2005). Dialogue act tagging for instant messaging chat sessions. In *Proceedings of the ACL Student Research Workshop*, (pp. 79–84). Association for Computational Linguistics.

Jeffreys, H. (1961). *Theory of Probability*, vol. 2 of *The International series of monographs on physics*. Oxford University Press.

Jekat, S., Klein, A., Maier, E., Maleck, I., Mast, M., & Quantz, J. (1995). Dialogue acts in VERBMOBIL.

Jurafsky, D., Ranganath, R., & McFarland, D. (2009). Extracting social meaning: Identifying interactional style in spoken conversation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, (pp. 638–646). Association for Computational Linguistics.

Jurafsky, D., Shriberg, E., & Biasca, D. (1997). Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. *Institute of Cognitive Science Technical Report*, (pp. 97–02).

Keizer, S., et al. (2002). Dialogue act recognition with Bayesian networks for Dutch dialogues. In *Proceedings of the 3rd SIGdial workshop on Discourse and dialogue-Volume 2*, (pp. 88–94). Association for Computational Linguistics.

Koopman, B. O. (1936). On Distributions Admitting A Sufficient Statistic. *Transactions of the American Mathematical Society*, *39*, 399–409.

Kottas, A. (2006). Dirichlet process mixtures of beta distributions, with applications to density and intensity estimation. In *Workshop on Learning with Nonparametric Bayesian Methods, 23rd International Conference on Machine Learning (ICML)*.

Kotz, S., Balakrishnan, N., & Johnson, N. L. (2000). *Continuous Multivariate Distributions*, vol. 1: Models of *Wiley series in probability and statistics*. John Wiley and Sons, 2nd ed.

Kreyszig, E. (2007). *Advanced engineering mathematics*. Wiley-India.

Kurihara, K., Welling, M., & Teh, Y. (2007). Collapsed variational Dirichlet process mixture models. In *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 20, (p. 19).

Kvam, P., & Day, D. (2001). The multivariate Polya distribution in combat modeling. *Naval Research Logistics*, *48*(1), 1–17.

Lee, J., Kim, G., & Seo, J. (1997). A dialogue analysis model with statistical speech act processing for dialogue machine translation. *Spoken Language Translation*, (pp. 10–15).

Lefebvre, M. (2006). *Applied probability and statistics*. Springer-Verlag New York Inc.

Lendvai, P., van den Bosch, A., & Krahmer, E. (2003). Machine learning for shallow interpretation of user utterances in spoken dialogue systems. In *Proc. of EACL-03 Workshop on Dialogue Systems: interaction, adaptation and styles of management*, (pp. 69–78).

Levin, L., Gates, D., Lavie, A., Pianesi, F., Wallace, D., Watanabe, T., & Woszczyna, M. (2000). Evaluation of a practical interlingua for task-oriented dialogue. In *NAACL-ANLP 2000 Workshop on Applied interlinguas: practical applications of interlingual approaches to NLP-Volume 2*, (pp. 18–23). Association for Computational Linguistics.

Levin, L., Langley, C., Lavie, A., Gates, D., Wallace, D., & Peterson, K. (2003). Domain specific speech acts for spoken language translation. In *Proceedings of 4th SIGdial Workshop on Discourse and Dialogue*.

Liscombe, J., Riccardi, G., & Hakkani-Tur, D. (2005). Using context to improve emotion detection in spoken dialog systems. In *Ninth European Conference on Speech Communication and Technology*.

Liu, J. (1996). Nonparametric hierarchical Bayes via sequential imputations. *The Annals of Statistics*, *24*(3), 911–930.

Loève, M. M. (1977). *Probability theory*. Springer, 4th ed.

MacEachern, S. (1994). Estimating normal means with a conjugate style Dirichlet process prior. *Communications in statistics. Simulation and computation*, *23*(3), 727–741.

MacEachern, S., & Muller, P. (1998). Estimating mixture of Dirichlet process models. *Journal of Computational and Graphical Statistics*, *7*(2), 223–238.

MacKay, D. (2003). *Information theory, inference, and learning algorithms*. Cambridge University Press.

MacKay, D. J. C. (1998). *Introduction to Gaussian Processes*, vol. 168 of *NATO ASI Series*, (pp. 133–166).

Madsen, R., Kauchak, D., & Elkan, C. (2005). Modeling word burstiness using the Dirichlet distribution. In *Proceedings of the 22nd international conference on Machine learning*, (pp. 545–552).

Mahmoud, H. M. (2009). Polya Urn Models. *Journal of the Royal Statistical Society Series A Statistics in Society*, *172*(4), 942–942.

Mairesse, F., & Walker, M. (2008). Trainable generation of big-five personality styles through data-driven parameter estimation. *ACL-08, Columbus*.

Mairesse, F., Walker, M., Mehl, M., & Moore, R. (2007). Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of Artificial Intelligence Research*, *30*(1), 457–500.

McAuliffe, J., Blei, D., & Jordan, M. (2006). Nonparametric empirical Bayes for the Dirichlet process mixture model. *Statistics and Computing*, *16*(1), 5–14.

McLachlan, G., & Peel, D. (2000). *Finite Mixture Models*, vol. 44 of *Wiley Series in Probability and Statistics*. Wiley-Interscience.

Meteopolis, N., & Ulam, S. (1949). The monte carlo method. *Journal of the American Statistical Association*, *44*(247), 335–341.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equations of state calculations by fast computing machines. *Journal of Medical Physics*, *21*(6), 1087–1092.

Miller, G. (2000). *The mating mind: How sexual choice shaped the evolution of human nature*. Doubleday Books.

Minka, T. (2001a). *A family of algorithms for approximate Bayesian inference*. Ph.D. thesis, Massachusetts Institute of Technology.

Minka, T. (2001b). Expectation propagation for approximate Bayesian inference. In *Uncertainty in Artificial Intelligence*, vol. 17, (pp. 362–369).

Minka, T. (2003). *Estimating a Dirichlet distribution*. Technical report, MIT.

Mukherjee, B., Zhang, L., Ghosh, M., & Sinha, S. (2007). Semiparametric Bayesian Analysis of Case–Control Data under Conditional Gene-Environment Independence. *Biometrics*, *63*(3), 834–844.

Nagata, M., & Morimoto, T. (1994). First steps towards statistical modeling of dialogue to predict the speech act type of the next utterance. *Speech Communication*, *15*(3-4), 193–203.

Neal, R. (2000). Markov chain sampling methods for Dirichlet process mixture models. *Journal of computational and graphical statistics*, *9*(2), 249–265.

Neal, R. M. (1993). *Probabilistic Inference Using Markov Chain Monte Carlo Methods*. Technical Report, Department of Computer Science, University of Toronto.

Pennebaker, J., Francis, M., & Booth, R. (2007). Linguistic inquiry and word count. *OperatorŠs Manual, The University of Texas at Austin*.

Pinker, S. (2003). Language as an adaptation to the cognitive niche. *Language evolution*, (pp. 16–37).

Pinker, S., Bloom, P., Barkow, J., Cosmides, L., & Tooby, J. (1992). Natural language and natural selection. *The adapted mind: Evolutionary psychology and the generation of culture*, (pp. 451–493).

Pitman, E. (1936). Sufficient statistics and intrinsic accuracy. *Proceedings Of The Cambridge Philosophical Society*, *32*, 567–579.

Pitman, J. (2006). *Combinatorial stochastic processes*, vol. 1875 of *Lecture Notes in Mathematics*. Springer-Verlag.

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

Power, R. (1979). The organisation of purposeful dialogues. *Linguistics*, *17*(1-2), 107–152.

Prasad, R., & Walker, M. (2002). Training a dialogue act tagger for human-human and human-computer travel dialogues. In *Proceedings of the 3rd SIGdial workshop on Discourse and dialogue-Volume 2*, (pp. 162–173). Association for Computational Linguistics.

Press, W., Flannery, B., Teukolsky, S., Vetterling, W., et al. (2007). *Numerical recipes*. Cambridge university press.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286.

Ranganath, R., Jurafsky, D., & McFarland, D. (2009). It's not you, it's me: Detecting flirting and its misperception in speed-dates. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, (pp. 334–342). Association for Computational Linguistics.

Ranganathan, A. (2006). The Dirichlet Process Mixture (DPM) Model.

Rasmussen, C. E., & Williams, C. (2006). Gaussian Processes for Machine Learning. *International Journal of Neural Systems*, *14*(2), 69–106.

Reithinger, N., & Klesen, M. (1997). Dialogue act classification using language models. In *Fifth European Conference on Speech Communication and Technology*.

Reithinger, N., & Maier, E. (1995). Utilizing statistical dialogue act processing in Verbmobil. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, (pp. 116–121). Association for Computational Linguistics.

Robert, C., & Casella, G. (2004). *Monte Carlo statistical methods*. Springer Verlag.

Rocktaeschel, O. (1922). Methoden zur Berechnung der Gammafunktion fur komplexes Argument. *University of Dresden, Dresden*.

Rodrigez, A. (2007). *Some Advances in Bayesian Nonparametric Modeling*. Ph.D. thesis, Duke University.

Rosenberg, A., & Hirschberg, J. (2005). Acoustic/prosodic and lexical correlates of charismatic speech. In *Ninth European Conference on Speech Communication and Technology*.

Rotaru, M. (2002). Dialog act tagging using memory-based learning. *Term project, University of Pittsburgh*.

Rude, S., Gortner, E., & Pennebaker, J. (2004). Language use of depressed and depression-vulnerable college students. *Cognition and Emotion*, *18*(8), 1121–1133.

Sacks, H., Schegloff, E., & Jefferson, G. (1974). A simplest semantics for the organization of turn-taking for conversation. *Language*, *50*(4), 696–735.

Sanchis, E., & Castro, M. (2002). Dialogue Act Connectionist Detection in a Spoken Dialogue System. *Soft Computing Systems. Design, Management and Applications*, *87*.

Schegloff, E. (1968). Sequencing in Conversational Openings1. *American anthropologist*, *70*(6), 1075–1095.

Searle, J. (1969). *Speech acts: An essay in the philosophy of language*. Cambridge university press.

Sethuraman, J. (1994). A constructive definition of Dirichlet priors. *Statistica Sinica*, *4*(2), 639–650.

Sherman, J., & Morrison, W. (1950). Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, *21*(1), 124–127.

Steyvers, M., & Griffiths, T. (2007). Probabilistic topic models. *Handbook of latent semantic analysis*, *427*.

Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Ess-Dykema, C., & Meteer, M. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, *26*(3), 339–373.

Strang, G. (2003). *Introduction to linear algebra*. Wellesley Cambridge Press.

Sudderth, E., Torralba, A., Freeman, W., & Willsky, A. (2006). Describing visual scenes using transformed dirichlet processes. *Advances in neural information processing systems*, *18*, 1297.

Sudderth, E. B. (2006). *Graphical Models for Visual Object Recognition and Tracking*. Ph.D. thesis, Massachusetts Institute of Technology.

Tamarit, V., Martinez-Hinarejos, C.-D., & Benedi, J.-M. (2011). On the use of n-gram transducers for dialogue annotation. In W. Minker, G. G. Lee, S. Nakamura, & J. Mariani (Eds.) *Spoken Dialogue Systems Technology and Design*, (pp. 255–276). Springer New York.

Teh, Y. W. (2006). A hierarchical Bayesian language model based on Pitman-Yor processes. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL ACL 06*, *44*(July), 985–992.

Teh, Y. W. (2010). Dirichlet processes. In *Encyclopedia of Machine Learning*. Springer.

Teh, Y. W., Jordan, M. I., Beal, M. J., & Blei, D. M. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, *101*(476), 1566–1581.

Traum, D. (1999). Speech acts for dialogue agents. *Foundations of rational agency*, (pp. 169–201).

Verbree, D., Rienks, R., & Heylen, D. (2006). Dialogue-act tagging using smart feature selection; results on multiple corpora. In *Spoken Language Technology Workshop, 2006. IEEE*, (pp. 70–73).

Wahlster, W. (1993). Verbmobil: Translation of face-to-face dialogs. In *In Proceedings of European Conference on Speech Communication and Technology*.

Wahlster, W. (2000). *Verbmobil: foundations of speech-to-speech translation*. Springer verlag.

Wainwright, M. J., & Jordan, M. I. (2008). Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, *1*, 1–305.

Wallach, H. M. (2008). *Structured Topic Models for Language*. Ph.D. thesis, University of Cambridge.

Walsh, B. (2004). Markov Chain Monte Carlo and Gibbs Sampling. *Lecture Notes*.

Webb, N. (2010). *Cue-Based Dialogue Act Classification*. Ph.D. thesis, University of Sheffield.

West, M. (1992). *Hyperparameter estimation in Dirichlet process mixture models*. Technical report 92-A03, Duke University.

West, M., & Escobar, M. (1993). Hierarchical priors and mixture models, with application in regression and density estimation. *Aspects of Uncertainty: A Tribute to D.V. Lindley*.

Winn, J. M. (2004). *Variational Message Passing and its Applications*. Ph.D. thesis, University of Cambridge.

Wright, H. (1998). Automatic utterance type detection using suprasegmental features. In *Proceedings of the International Conference on Spoken Language Processing*, vol. 4, (pp. 1403–1406).

Yang, X. (2008). *Introduction to computational mathematics*. World Scientific Publishing.

Zhang, X. (2008). A Very Gentle Note on the Construction of Dirichlet Process. *The Australian National University, Canberra*.