Latent Dirichlet Co-Clustering

M. Mahdi Shafiei and Evangelos E. Milios
Faculty of Computer Science, Dalhousie University
6050 University Ave., Halifax, Canada
shafiei@cs.dal.ca , eem@cs.dal.ca

Abstract

We present a generative model for simultaneously clustering documents and terms. Our model is a four-level hierarchical Bayesian model, in which each document is modeled as a random mixture of document topics *, where each topic is a distribution over some segments of the text. Each of these segments in the document can be modeled as a mixture of* word topics *where each topic is a distribution over words. We present efficient approximate inference techniques based on Markov Chain Monte Carlo method and a Moment-Matching algorithm for empirical Bayes parameter estimation. We report results in document modeling, document and term clustering, comparing to other topic models, Clustering and Co-Clustering algorithms including Latent Dirichlet Allocation (LDA), Model-based Overlapping Clustering (MOC), Model-based Overlapping Co-Clustering (MOCC) and Information-Theoretic Co-Clustering (ITCC).*

1 Introduction

Finding the appropriate representation model for text data has been one of the main issues for the data mining community since it started to look at the problem of processing text automatically. The "bag-of-words" representation is the basic and most widely used representation method for textual data [19]. In this approach, the order of words at which they appear in documents are ignored and only the word frequencies are taken into account. But this approach has been criticized for several reasons. Among those, it provides a relatively high dimensional representation of data (equal to the dictionary size) which causes curse of dimensionality problem [19]. Furthermore, it does not consider synonymy and polysemy relations of words in natural language. It has been also criticized of losing information due to its ignorance of word order. Various preprocessing steps such as removing *stop-words* and stemming have been used to reduce dimensionality, create and select better features.

To overcome the high dimensionality issue of the bag-of-words representation, several dimension reduction methods have been proposed. Feature selection methods select a subset of words to reduce the dimensionality. Feature transformation methods try to tackle not only the high dimensionality problem of "bag-of-words" representation, but indirectly consider synonymy and polysemy as well. Latent Semantic Indexing (LSI) [6] is one of these approaches which uses singular value decomposition to identify a linear subspace in the original space of features. It is believed that the resulting new features also capture the two mentioned properties of natural language - polysemy and synonymy.

But the problem with most cartesian space representation approaches for text like LSI is their inability to provide interpretable components. Despite some work on interpreting the dimensions generated by these methods [5], these approaches are still far from providing a natural interpretation in the case of text. Topic models, on the other hand, are a class of statistical models in which the semantic properties of words and documents are expressed in terms of probabilistic topics. Probabilistic topic modeling as a way of representing the content of words and documents has the distinct advantage that each topic is individually interpretable, providing a probability distribution over words that picks out a coherent cluster of correlated terms. The major difference between cartesian space methods like LSI and statistical topic models is that LSI family methods claim that words and documents can be represented as points in the Euclidean space whereas for the topic models, this is not the case.

One common assumption among most statistical models for language is still the *bag-of-words assumption*. In these models, no assumption is made about the order of words. In other words, while this family of methods tries to deal with the two first issues of bag-of-words representation, high dimensionality and ignoring polysemy and synonymy properties, it still keeps the "bag-of-words" assumption intact. Recently, there has been increased research interest in models sensitive to this kind of information [11].

Proceedings of the Sixth International Conference on Data Mining (ICDM'06)
0-7695-2701-9/06 $20.00 © 2006 IEEE

The basic idea behind all proposed topic models [10, 3] is that a document is a mixture of several topics where each topic is some distribution over words. Each topic model is a generative model which specifies a simple probabilistic process by which the words in a document are being generated on the basis of a small number of latent variables.

Using standard statistical techniques, one can invert the process and infer the set of latent variables responsible for generating a given set of documents [21]. Assuming a model for generating the data, the goal of fitting this generative model is to find the best set of latent variables that can explain the observed data (i.e., observed words in documents).

Probabilistic Latent Semantic Analysis (also known as the aspect model) [12] was one of the first attempts toward using probabilistic models for document and text modeling. In this model, each word is assumed to be a sample from a mixture model. Mixture components are multinomial random variables that can be viewed as representations of "topics". Each word is generated from a single topic and a document is a collection of words generated potentially from different topics.

Though a useful step after LSI, the PLSI model does not provide a generative model for a document, instead it is a model for word/document co-occurrences [13]. This assumption makes it difficult to assign probabilities to documents outside of the learning corpus. Latent Dirichlet Allocation [3], on the contrary, is a true generative model for documents and therefore provides the means for generating both the observed and unseen documents. In LDA, the documents are assumed to be sampled from a random mixture over latent topics, where each topic is characterized by a distribution over words. Furthermore, the mixture coefficients are also assumed to be random and by considering a prior probability on them, LDA provides a complete generative model for the documents [10].

In Latent Dirichlet Allocation, a document is generated by first picking a distribution over latent topics from a Dirichlet distribution, which determines the multinomial distribution over topics for words in that document. The words in the document are then generated by picking a topic for each word from this distribution and then picking a word from that topic according to another multinomial distribution. Fig. 1.b shows the graphical model corresponding to the generative model of LDA.

The major and direct output of these models is a set of overlapping clusters of words. Clustering documents can be viewed only as a byproduct and not as a direct output of topic models. On the other hand, co-clustering [8, 20, 15] is a data mining technique with various applications such as text clustering and microarray analysis. Co-clustering algorithms try to simultaneously cluster rows and columns of a two-dimensional data matrix. One of the benefits of co-clustering algorithms is taking advantage of the duality between documents and words and in general the duality between the rows and columns of an adjacency matrix. Co-clustering algorithms, using the clustering results on words as a low dimensional representation of documents can achieve a more accurate clustering for documents [8]. In this work, we try to combine these two ideas, probabilistic topic models and co-clustering, using topic models to construct a low-dimensional representation of documents. Therefore, we extend the original idea of topic models to consider the resulting low-dimensional representation of documents in another nested topic model for clustering documents.

The topics discovered by most probabilistic topic models capture the correlation between words, but the correlations between topics are not modeled. Several models have been recently proposed to capture the correlation between topics, such as Hierarchical Dirichlet Processes Model (HDP) [22], Correlated Topic Models (CTM) [2] and Pachinko Allocation Model (PAM) [14]. In natural text data, it is common to have correlations among topics. As pointed out in [2], "a document about sports is more likely to also be about health than international finance". In the LDA model, the topic proportions are derived from a Dirichlet distribution and hence are nearly independent. CTM tries to capture topic correlations by introducing logistic normal distribution instead of Dirichlet distribution for drawing topic mixture proportions. The logistic normal distribution is yet again a distribution on the simplex where the correlation between pairs of components is described through a covariance matrix. In CTM, only the pairwise correlations are modeled and the number of parameters grows quadratically with the number of topics [14]. In the PAM model, similar to our proposed model, the concept of topic is extended to include not only distributions over words, but also distribution over topics. The model structure is an arbitrary DAG where each leaf is associated with a word and each non-leaf node is a distribution over its children. The direct parents of the leaf nodes are distributions over words and correspond to topics in LDA. All other interior nodes are distributions over topics and called "super-topics". Allowing an arbitrary DAG structure, the PAM model is able to capture arbitrary correlations between word topics.

In this paper, we propose a generative model for text documents based on LDA model which is able to cluster both words and documents simultaneously. The model is also more sensitive to to the locations of words in documents by focusing on meaningful segments of text. This enables the model to detect multiple topics covered by a document.

The rest of this paper is organized as follows. In section 2, we present the Latent Dirichlet Co-Clustering Model (LDCC). We propose inference and parameter estimation algorithms for LDCC in Section 3. Finally, we conclude the

paper with a brief review of the paper and some discussion on future works in section 5.

2 Latent Dirichlet Co-Clustering Model

We use the same notation and definitions as in [3]. We define the following terms:

- A *word* is the basic building block of our data and it is selected from a vocabulary indexed by $\{1, \ldots, V\}$. We represent words using unit-basis vectors that have a single component equal to one and all other components equal to zero. Thus, the vth word in the vocabulary is represented by a V-vector w such that the vth component is one and all other components are zero.

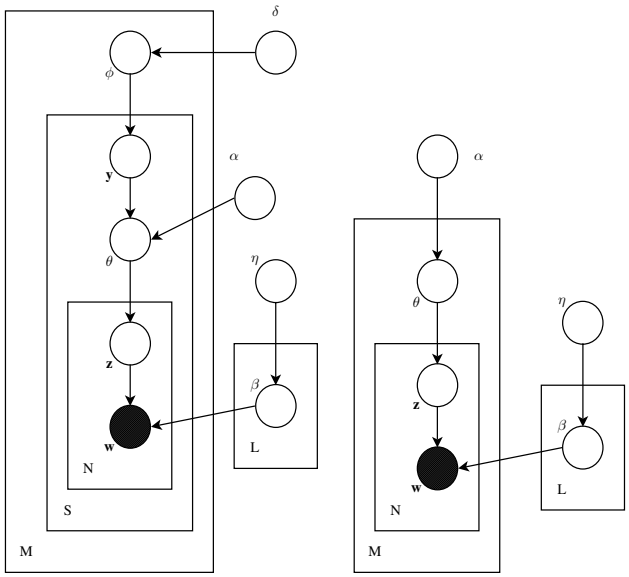
- A *document* is a sequence of words $\mathbf{w} = (w_1, w_2, \ldots, w_N)$, where N is the number of words in the document.

- A *corpus* is a collection of M documents denoted by $\mathcal{D} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_M\}$

The basic idea of the LDA model is to assume each document as a random mixture over latent topics, where each topic is specified by a distribution over words. We extend this idea by assuming each document is a random mixture of topics , where each topic is a distribution over some segments of the document. Now, each of these segments in the document can be modeled by LDA.

The intuition behind this work is that documents are composed of meaningful single-topic segments put together. Each of these segments is assumed to convey a single concept or topic. This topic is among a handful of topics which specifies the theme of the document. If one looks at each of these segments separately, the order of words in the segment is assumed to have little impact on the concept which the segment is trying to convey. Thus, the "bag-of-words" assumption for these segments is fairly realistic, unlike for the whole document. In this work, we assume segments are paragraphs of the text. The proposed model tries to model each segment based on its word content similar to most probabilistic topic models. Then these learned topics on the words are being used to represent document topics. In other words, each document topic is considered a mixture of word topics where the mixture coefficients uniquely specifies the document topic.

The generative probabilistic model we propose is shown as a graphical model in Fig. 1. Plate notation [4] is a standard and convenient way of illustrating probabilistic generative models with repeated sampling steps. In this graphical notation, shaded and unshaded variables indicate observed and latent (i.e., unobserved) variables respectively.



(a) Co-Clustering Model (b) LDA Model

Figure 1. LDA Model and Co-Clustering Model inspired by LDA

The LDCC model assumes the following generative process for each document d in a corpus D (intuitive explanations of model parameters are given in the text following the overview of the generative process.):

1. Choose $S \sim Poisson(\mu)$: number of segments (paragraphs or sentences) in the document
2. Choose $\phi \sim Dir(\delta)$
3. For each of the S segments s
   (a) Choose a topic for the segment $y_s \sim Multinomial(\phi)$
   (b) Choose $N_s \sim Poisson(\varepsilon)$: number of words in the segment
   (c) Choose $\theta_s \sim Dir(\alpha, y_s)$
   (d) For each of the N_s words w_{sn}
      i. Choose a topic $z_{sn} \sim Multinomial(\theta_s)$
      ii. Choose a word w_{sn} from $P(w_{sn}|z_{sn}, \beta)$, a multinomial probability conditioned on the topic z_{sn}

We have assumed that the number of word and document topics (and hence the dimensionality of topic variables z and y) are known and fixed. We also model the word probabilities conditioned on the topics by a $L \times V$ matrix β where $\beta_{ij} = p(w^j = 1|z^i = 1)$ which is assumed to be fixed and will be estimated though the learning process. Finally, as in the LDA model, we can use any other document length

distribution instead of the Poisson distribution as it is not important for the rest of the model. Furthermore, note that the N_s variables are independent of all the other data generating variables (θ, z, ϕ and y) and we therefore ignore its randomness in the subsequent development.

Note that ϕ represents the mixing proportion of document-topics in a document. It specifies the parameters of the K-dimensional multinomial distribution from which the model draws samples for document topics. θ_s is a sample from the Dirichelt distribution and specifies the mixing proportion of word-topics in the text segment s. Note that this mixing proportion depends on the document-topic that the current text segment is generated from. The model assume that each document-topic is a mixture of several word-topics and this fact is modeled through the matrix of hyperparameters α.

The Dirichlet distribution is a conjugate prior for the multinomial distribution. Choosing a conjugate prior makes the problem of statistical inference easier. Basically, the posterior distribution would have the same functional form as prior distribution and the process of statistical inference, instead of being caught up in impractical integrations, be reduced to simply adjusting the parameters of the posterior distribution given the new evidence.

A k-dimensional Dirichlet random variable θ can take values in the $(k-1)$-simplex. The probability density of a k-dimensional distribution on this simplex is defined by:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k}\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\theta_1^{\alpha_1-1}\ldots\theta_k^{\alpha_k-1} \qquad (1)$$

The parameters of this distribution are represented by a k-vector α with components $\alpha_i > 0$. Each hyperparameter α_i has the nice interpretation that it could be considered as a prior observation count of the number of times that the corresponding topic has been sampled. $\Gamma(x)$ is the Gamma function.

Given the parameters α, β and γ, the joint distribution of a word-topic mixture θ, document-topic mixture ϕ, a set of N word-topics $\mathbf{z}$, a set of S document-topics $\mathbf{y}$, and a set of $N \times S$ words $\mathbf{w}$ is given by:

$$\begin{aligned} p(\phi,\mathbf{y},\theta,\mathbf{z},\mathbf{w}|\alpha,\beta,\delta) &= p(\phi|\delta)\prod_{s=1}^{S}p(y_s|\phi)p(\theta_s|\alpha,y_s) \\ &\quad \prod_{n=1}^{N_s}p(z_{sn}|\theta_s)p(w_{sn}|z_{sn},\beta) \end{aligned}$$

where $p(z_{sn}|\theta_s)$ is simply θ_{si} for unique i such that $z_{sn}^i = 1$ and $p(y_s|\phi)$ is simply ϕ_i for unique i such that $y_s^i = 1$. Note that variables z_{sn} and y_s are boolean vectors that have a single component equal to one and all other components zero. The component equal to one simply represents the word-topic or document-topic that the corresponding word or segment belongs to. Integrating over θ and ϕ and summing over z and y, we obtain the marginal distribution of a document $\mathbf{w}$:

$$p(\mathbf{w}|\alpha,\beta,\delta) = \int p(\phi|\delta)\left(\prod_{s=1}^{S}\sum_{y_s}p(y_s|\phi)\int p(\theta_s|\alpha,y_s)\left(\prod_{n=1}^{N_s}\sum_{z_{sn}}p(z_{sn}|\theta_s)p(w_{sn}|z_{sn},\beta)\right)d\theta_s\right)d\phi$$

Taking the product of marginal probabilities of documents in a corpus gives us the probability of the corpus.

$$p(\mathcal{D}|\alpha,\beta,\delta) = \prod_{d=1}^{M}p(\mathbf{w}_d|\alpha,\beta,\delta)$$

3 Inference and Parameter Estimation

The inference problem is to compute the posterior distribution of hidden variables given the input variables α,β,δ and observations $\mathbf{w}$:

$$p(\phi,\mathbf{y},\theta,\mathbf{z}|\mathbf{w},\alpha,\beta,\delta) = \frac{p(\phi,\mathbf{y},\theta,\mathbf{z},\mathbf{w}|\alpha,\beta,\delta)}{p(\mathbf{w}|\alpha,\beta,\delta)}$$

which is intractable to compute in general. Given a document collection, we also need to estimate the model parameters α,η,δ so that the model likelihood for the collection gets maximized.

Exact inference on models in the LDA family cannot be performed practically. Three standard approximation methods have been used to carry out the inference and obtain practical results: variational methods [3], Gibbs sampling [10], and expectation propagation [16]. The EM based algorithms tend to face local maxima problems in this models [3]. Therefore, we use algorithms in which some of the hidden parameters - in our case β,ϕ and δ - can be integrated out instead of explicitly being estimated. Note that we use conjugate priors in our model, and thus we can easily integrate out these parameters. This simplifies the sampling since we do not need to sample β,ϕ and δ at all. Besides doing inference for document and word topic assignment variables y and z, we also need to learn the parameters of the Dirichlet distribution $\alpha = \{\alpha_1,\ldots,\alpha_K\}$. In this section, we describe procedures for inference and parameter estimation and present a Gibbs sampling procedure for doing inference in the proposed model.

MCMC algorithms are a family of approximate iterative algorithms used to draw samples from a complex and usually high-dimensional distribution. Gibbs sampling is a member of this family and is applicable where the whole joint distribution is unknown or impractical to sample from, but the conditional distributions are known and sampling from them is not difficult.

In each turn of the algorithm, a subset of variables are sampled from their conditional distribution conditioned on the current values of all other variables. This process is performed sequentially and continues until the sampled values approximate the target distribution. In our problem, the distribution that we want to sample from is the posterior distribution of word-topics and document-topics given the collection of documents. Since this distribution is intractable and difficult to sample from, in each iteration of Gibbs sampling, we sample from the conditional distribution of a single word in a document given that the topic assignment for all other words and paragraphs in all documents except the current word are known. We also sample from the conditional distribution of a single paragraph given that the topic assignments of all other words not in the current paragraph and topic assignments of all other paragraphs are known. For our proposed model, Gibbs sampling algorithm is easy to implement, requires little memory, and is competitive in speed and performance with existing algorithms.

We order the documents in the corpus and represent the collection of documents by three list of indices: word indices wl, paragraph indices pl and document indices dl. wl_i denotes the index of the ith word in the sequence of words (if we assume the whole corpus as a sequence of words fed to the algorithm) and dl_i and pl_i are the document index and paragraph index of the corresponding word respectively. These lists will then be fed to the Gibbs Sampling algorithm. For each word token, the Gibbs sampling algorithm estimates the probability of assigning the current word to word-topics given assignment of all other words to word-topics from the corresponding conditional distribution that we will derive shortly. Then the current word would be assigned to a word-topic and this assignment will be stored for being used when the Gibbs sampling algorithm works on other words.

While scanning the list of words, we keep track of the paragraphs. For each new paragraph, the Gibbs sampling algorithm estimates the probability of assigning this paragraph to document-topics given assignments of all other paragraphs to document-topics. These probabilities are computed from the corresponding conditional distribution for a paragraph given all other topic assignment to every other paragraph and all words not in this paragraph. Then the new paragraph would be assigned to a document-topic.

In our case we need to compute the conditional distribution $p(z_{dsn}|z_{-dsn}, y, w)$ and $p(y_{ds}|z, y_{-ds}, w)$, where z_{dsn} represents the word-topic assignment for word w_{dsn} (word n in document d and paragraph s) and z_{-dsn} denotes the word-topic assignments for all other words except the current word w_{dsn}. y_{ds} denotes the document-topic assignment for paragraph p_{ds} in document d and y_{-ds} represents the document-topic assignments for all paragraphs except the current paragraph p_{ds}. Beginning with the joint probability of a dataset, and using the chain rule, we can obtain the conditional probabilities conveniently. The derivations are provided in detail in Appendix A. For the LDCC Model, we obtain:

Algorithm 1: LDCC Gibbs Sampling Algorithm

Input: δ, α, η, L, K,Corpus, $MaxIteration$
Output: topic assignments for all words and paragraphs in the Corpus

Initialization: Randomly, initialize the word-topic and document topic assignments for all word token and paragraphs
Compute P_{dk} for all values of $k \in \{1..K\}$ and all documents
Compute n_{lv} for all values of $l \in \{1..L\}$ and all word tokens
Compute $n_l^{(ds)}$ for all values of $l \in \{1..L\}$ and all documents and their paragraphs
if *doing parameter estimation* **then**
    Initialize $alpha$ parameters using Eq. 4
Randomize the order of documents in the corpus
Randomize the order of paragraphs in each document
Randomize the order of words in each paragraph
for $iter \leftarrow 1$ **to** $MaxIteration$ **do**
    foreach *word i according to the order* **do**
        Exclude word i and its assigned topic l from variables $n_l^{(ds)}$ and n_{li}
        $newl =$ sample new word-topic for word i using Eq. 2
        Update variables $n_l^{(ds)}$ and n_{li} using the new word-topic $newl$ for word i
        if *entered a new paragraph j* **then**
            Exclude paragraph j and its assigned topic k from variable P_{dk}
            $newk =$ sample new document-topic for paragraph j using Eq. 3
            Update variable P_{dk} using the new document-topic $newk$ for paragraph j
            if *doing parameter estimation* **then**
                Update $alpha$ parameters using Eqs. 4

$$p(z_{dsn}|z_{-dsn}, y, w) = (\alpha_{y_{ds} z_{dsn}} + n^{(ds)}_{z_{dsn}} - 1) \times \frac{n_{z_{dsn} w_{dsn}} + \eta_{w_{dsn}} - 1}{\sum_{v=1}^{V} n_{z_{dsn} v} + \eta_v - 1} \quad (2)$$

where $n^{(ds)}_{z_{dsn}}$ represents how many times a word in paragraph s of document d has been assigned to topic z_{dsn}. Furthermore, $\alpha_{y_{ds} z_{dsn}}$ is the z_{dsn}th component in $\alpha_{y_{ds}}$. $n_{z_{dsn} w_{dsn}}$ represents the total number of times that the word w_{dsn} has been assigned to topic z_{dsn}. For $p(y_{ds}|z, y_{-ds}, w)$, we have

$$p(y_{ds}|z, y_{-ds}, w) = (\delta_{y_{ds}} + P_{d y_{ds}} - 1) \times \frac{\prod_{l=1}^{L} \prod_{j=0}^{n_l^{(ds)}-1} (\alpha_{y_{ds} l} + j)}{\prod_{j=0}^{n^{(ds)}-1} (\sum_{l=1}^{L} \alpha_{y_{ds} l} + j)} \quad (3)$$

where y_{ds} is the document-topic that has been assigned to paragraph s in document d and $P_{d y_{ds}}$ is the number of times a paragraph in document d has been assigned to document-topic y_{ds}. $n^{(ds)}$ is the number of words in paragraph s of

SYMBOL	DESCRIPTION
L	number of word-topics
K	number of document-topics
N_s	number of words in paragraph s
M	number of documents in the collection
$\mathbf{y}$	document-topic variable for the corpus
$\mathbf{z}$	word-topic variable for the corpus
δ	parameters of the Dirichlet prior on document-topics
α	matrix of $K \times L$ dimensions, row i represents mixing proportion of word-topics in document-topic i
β	parameters of multinomial distribution of words conditioned on word-topics
η	parameters of the prior probability for distribution of words conditioned on word-topics
ϕ	mixing proportion of document-topics in document
θ_s	mixing proportion of word-topics in the text segment s
w_{dsn}	word n in paragraph s of document d
z_{dsn}	word-topic assignment for word w_{dsn}
z_{-dsn}	word-topic assignments for all other words except the current word w_{dsn}
n_{lv}	number of times that the word v assigned to topic l
$n_l^{(ds)}$	number of times a word in paragraph s of document d assigned to topic l
y_{ds}	document-topic assigned to paragraph s in document d
P_{dk}	number of paragraphs in document d assigned to document-topic k
$n^{(ds)}$	number of words in paragraph s of document d

Table 1. List of symbols used in this paper

document d. $\delta_{y_{ds}}$ is the corresponding Dirichlet parameter for document-topic y_{ds} that the paragraph s in document d has been assigned to.

The Gibbs sampling algorithm is initialized by assigning each word token to a random word-topic in $[1..L]$ and each paragraph to random document-topic $[1..K]$. A number of initial samples have to be discarded (also known as burn-in samples) because they are poor estimates of the posterior. After this burn-in period, the next Gibbs samples start to approximate the target distribution (i.e., the posterior distribution over word-topic and document-topic assignments). Now, we pick a number of Gibbs samples and save them as a representative set of samples from this distribution. This should be done at regularly spaced intervals to prevent correlations between samples [9].

In the LDA model as adopted by previous works, the Dirichlet parameters α are assumed to be given and fixed. This would give us reasonable results when we choose a uniform Dirichlet. But for our proposed model, the parameters α capture relationships between document and word topics and must be learned from the data. In a sense, they somehow summarize the corresponding term-document matrix of the corpus. For estimating parameters of a Dirichlet distribution, one can use different approaches proposed in the literature [17]. These methods are based on maximum likelihood or maximum a posteriori estimation of parameters. There is no closed-form solution for these methods and one should use iterative methods to learn the parameters. In order to avoid these often computationally expensive methods, we use moment matching [17] to approximate the parameters of the Dirichlet prior α. In each iteration of Gibbs sampling, we update

$$\begin{aligned} mean_{kl} &= \frac{1}{N_k} \sum_{s \in S_k} \frac{n_l^{(s)}}{n^{(s)}} \\ var_{kl} &= \frac{1}{N_k} \sum_{s \in S_k} (\frac{n_l^{(s)}}{n^{(s)}} - mean_{kl}) \\ m_{kl} &= \frac{mean_{kl}(1 - mean_{kl})}{var_{kl}} - 1 \\ \alpha_{kl} &\propto mean_{kl} \\ \sum_{l=1}^{L} \alpha_{kl} &= \exp(\frac{\sum_{l=1}^{L} \log(m_{kl})}{L-1}) \end{aligned} \quad (4)$$

where S_k represents the set of paragraphs assigned to document-topic k and N_k is the number of paragraphs assigned to document-topic k. $n_l^{(s)}$ represents the number of times a word in paragraph s has been assigned to word-topic l. $n^{(s)}$ is the number of words in paragraph s. Note that for $mean_{kl}$ and var_{kl}, we only consider the paragraphs assigned to document-topic k. For each document-topic k, we first compute sample mean $mean_{kl}$ and sample variance var_{kl}. They are computed over all paragraphs assigned to document-topic k. Algorithm 1 shows the pseudocode for the Gibbs sampling process for the proposed model. A summary of symbols and their description is given in Table 1.

4 Experimental Results

We use two real-world datasets in our experiments. We built our first dataset using NIPS conference papers available in both text and XML format [1]. Each paper corresponds to a XML file contains nested tags for pages, columns, paragraphs, lines, and words. We removed all the words occurred in less than 5 documents from the list of final word tokens. We also used a list of standard "stop-words" and deleted all numbers, words with length less than 3 and having non-ascii characters. For NIPS dataset, we keep certain two characters length words like "EM" and "ML". We do not consider paragraphs with less than 5 words and do not also include documents with less than 3 paragraphs. As a result, the NIPS dataset contains 1803 documents with the total of 1858577 word tokens. There are 11891 paragraphs in this dataset and word tokens are taken from 20485 unique words.

Our second dataset is a subset of Wikipedia XML corpus [2] [7]. This subset contains 1236 articles categorized in 9 overlapping classes. Each documents belongs to

[1] It is available at http://nips.djvuzone.org/txt.html

[2] it is available for download at http://www-connex.lip6.fr/~denoyer/wikipediaXML by registration

topic 1	topic 2	topic 3	topic 4	topic 5	topic 6	topic 7	topic 8	topic 9
error	neuron	image	analog	data	control	function	rule	distribution
generalization	neurons	images	circuit	clustering	model	functions	rules	probability
learning	synaptic	object	current	principal	motor	basis	set	gaussian
training	firing	recognition	figure	cluster	forward	linear	step	data
optimal	spike	face	chip	pca	inverse	regression	form	parameters
order	time	objects	voltage	set	dynamics	kernel	fuzzy	model
large	activity	hand	vlsi	algorithm	controller	space	problem	bayesian
average	rate	pixel	circuits	points	feedback	gaussian	relative	mixture
small	synapses	system	digital	approach	system	approximation	extraction	density
examples	potential	view	implementation	clusters	position	rbf	expert	likelihood

Figure 2. Example word-topics for the NIPS dataset

topic 1	topic 2	topic 3	topic 4	topic 5	topic 6	topic 7	topic 8	topic 9	topic 10
language	game	church	house	air	league	war	apollo	party	system
english	player	god	parliament	aircraft	football	german	earth	government	computer
greek	cards	christian	members	world	team	army	moon	president	game
languages	players	jesus	commons	force	world	soviet	lunar	political	games
word	games	christ	lords	military	club	battle	time	national	apple
russell	play	orthodox	bill	ship	home	germany	mission	minister	atari
century	card	baptism	act	gun	season	world	program	states	commodore
theory	hand	life	power	war	won	forces	module	united	home
words	round	catholic	chopin	ships	game	french	jpg	election	software
modern	played	roman	speaker	navy	major	union	crew	state	video

Figure 3. Example word-topics for the Wikipedia dataset

1.47 classes on average. The biggest class corresponds to "Art/Categories" whit 510 documents. The smallest class, "United Kingdom/Categories" has 74 documents. There are 774958 word tokens, 21453 paragraphs and 17406 unique words after preprocessing. The preprocessing phase is similar to the one for NIPS dataset. For the Wikipedia dataset, we do not have the tags for separating words, therefore we used all delimiting characters to separate words.

In this section, we describe the details of our experiments that demonstrate the improved performance of LDCC on NIPS dataset, compared to the LDA in terms of generalization of the topics found measured by perplexity. We also show the improved clustering performance of LDCC compared to the MOC and MOCC models.

In Gibbs sampling for both LDCC and LDA, we run 5 markov chains, discarding the first 500 iterations as burn-in iterations, and then draw 5 samples from each chain at a lag of 50 iterations, a total of 25 samples for each experiment. For the NIPS dataset, the total training time for LDCC is approximately 23 hours on a machine with a dual core Intel Pentium IV 64-bit ($EM64T$) processor (2×3.0GHz processor) with 2GB of RAM.

4.1 Word and Document Topic Examples

In this section, we show 9 word-topics derived from NIPS dataset and 10 word-topics derived from Wikipedia dataset, each represented by their first 10 most probable words, presented in Fig.2 and Fig.3 respectively. As it can be seen, the model seems to be able to capture some of the underlying word-topics in both datasets.

We have also constructed a graph of latent word-topics and document-topics which explains the correlations found by the model amongst word-topics appearing in a cluster of documents represented by a document-topic. A part of this graph is shown in Figure 4. For each word-topic in the graph, we have a box where the word-topic is represented by its 6 most probable words. For each document-topic k, we rank the word-topics $\{l\}$according to Dirichlet parameters α_{kl}. From the top 10 word-topics for each document-topics, we have picked some of them and it is depicted in Figure 4. The idea is that we try to get word-topics as tight as possible representing a very specific word-topic. We can illustrate these word-topics by a set of their most probable words. Document-topics are distributions over these word-topics and thus theoretically can be represented by their most probable words. But each document-topic can be a mixture of seemingly unrelated word-topics and this makes representation of a document-topic with words less descriptive than with word-topics. Representing document-topics with a set of most probable word-topics would allow the user himself to figure out the associated concept. Additionally, this representation makes the visualization of word-topic correlations more intuitive.

4.2 Likelihood Comparison for Document Modeling

We trained LDA and LDCC using the training set and we want to compare the generalization performance of these two models in terms of the likelihood achieved on the test set. Perplexity is a widely used and standard measure for comparing the performance of statistical models for natural language [3]. Perplexity can be though of as the uncer-

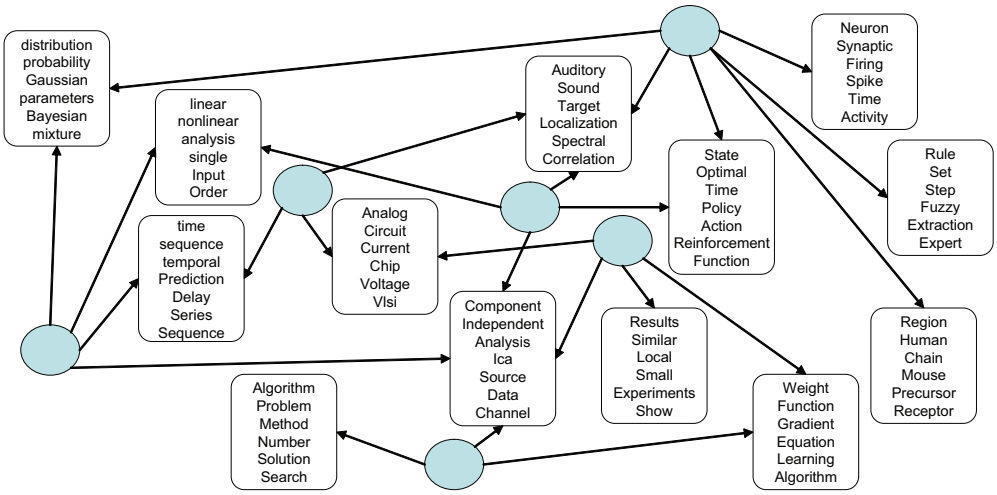


Figure 4. Correlation identified by LDCC between word-topics. Each circle shows a document-topic and each box corresponds to a word-topic. As it can be seen, one document-topic can be connected to several word-topics and capture their correlation.

tainty in predicting a single word according to the model and lower values are better. Formally, for a test set of M documents, it is defined as:

$$perplexity(\mathcal{D}_{test}) = \exp\left(-\frac{\sum_{d=1}^{M}\log p(\mathbf{w}_d)}{\sum_{d=1}^{M} N_d}\right) \quad (5)$$

In order to compute perplexity, we need to compute the likelihood $p(\mathbf{w})$ and this requires summing over all possible assignments of words to *word topics* $\mathbf{z}$ and text segments (in our datasets, the text segments corresponds to paragraphs) to *document topics* $\mathbf{y}$. This problem has no closed-form solution. Previous work on LDA [10] has used harmonic mean estimator introduced in [18]. We estimate $p(\mathbf{w})$ by taking the harmonic mean of a set of values $p(\mathbf{w}|\mathbf{z})$. By using the chain rule and integrating the parameter out, we get:

$$p(\mathbf{w}|\mathbf{z}) = \left(\frac{\Gamma(\sum_{v=1}^{V}\eta_v)}{\prod_{v=1}^{V}\Gamma(\eta_v)}\right)^{L}\prod_{l=1}^{L}\frac{\prod_{v=1}^{V}\Gamma(n_{lv}+\eta_v)}{\Gamma(\sum_{v=1}^{V}n_{lv}+\eta_v)} \quad (6)$$

$\mathbf{z}$ is sampled from the posterior $P(\mathbf{z}|\mathbf{w})$ using the Gibbs sampling procedure described in section 3. The derivation of this is similar to the one described in Appendix A.

In these experiments, we use the NIPS dataset and split it into 80% for training and 20% for calculating the likelihood. We use 20 *document topics* and change the number of *word topics* from 20 to 100. We split the dataset randomly so that the training and test subsets have relatively 80% and 20% of the documents respectively and each topic in both subsets has at least 5 documents.

We present perplexity results for different number of word topics in Fig. 5. For all these experiments, the number of document topics are assumed fixed and equal to 20. As it can be observed, for different number of word-topics,

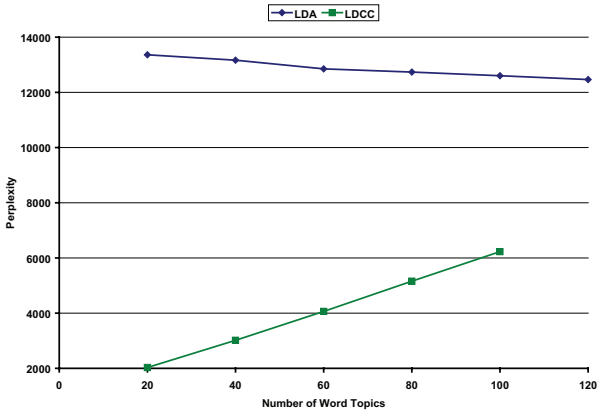


Figure 5. Perplexity results for NIPS dataset with different numbers of topics

LDCC always produce lower perplexity compared to LDA. As it can be seen in Fig. 5, the perplexity for the LDCC method is increasing in the range of values that we have examined, as opposed to LDA model. This shows that the proposed method can not keep up its generalization performance as the number of word-topics increases, in contrast to the LDA model.

We also show the results comparing LDCC and LDA when we use different amount of training data for learning the model parameters. In these set of experiments, the number of word-topics and document-topics are assumed fixed and equal to 20 and 100 respectively. We present these results in Fig. 6. As it can be seen, the LDCC model has lower perplexity compared to LDA model as the amount of training data increase. The perplexity for LDCC model decreases while we increase the amount of training data while for the LDA model, the perplexity has a peak when we use

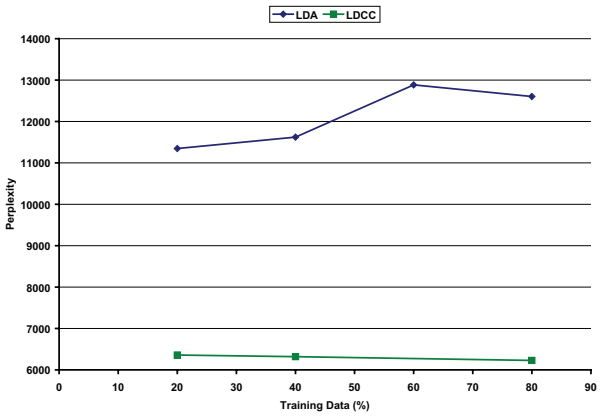


Figure 6. Perplexity results for NIPS dataset with different amounts of training data

60% of the data for training.

4.3 Document Clustering Performance

We compare LDCC algorithm in terms of clustering accuracy with another algorithm for overlapping clustering, namely Model-based Overlapping Clustering (MOC) [1] and two other co-clustering algorithms, namely Model-based Overlapping Co-Clustering (MOCC) [20] and Information-Theoretic Co-Clustering (ITCC) [8]. We conducted this experiment on our subset of Wikipedia XML corpus [7].

In order to compare clustering results, we use precision, recall, and F-measure calculated over pairs of points, as defined in [1]. For each pair of points that share at least one cluster in the overlapping clustering results, these measures try to estimate whether the prediction of this pair as being in the same cluster was correct with respect to the underlying true categories in the data. Precision is calculated as the fraction of pairs correctly put in the same cluster, recall is the fraction of actual pairs that were identified, and F-measure is the harmonic mean of precision and recall.

Table 2 presents the results of LDCC versus ITCC algorithm in terms of precision, recall and F-Measure for the Wikipedia Corpus. Each reported result is an average over ten trials. We have chosen the number of word-topics to be fixed and equal to 50. Table 2 contains the results for two different values for the number of document-topics, 15 and 20. Table 2 shows that the precision of LDCC is very close to the other three methods investigated but it can also be seen that our proposed algorithm shows a major improvement in terms of recall and F-Measure.

Algorithm	K	Precision	Recall	F-Measure
LDCC	15	30.88	**79.21**	**44.43**
MOC	15	31.75	42.45	36.33
MOCC	15	31.30	70.06	43.26
ITCC	15	31.06	7.44	12.00
LDCC	20	31.10	**75.70**	**44.09**
MOC	20	31.84	48.06	38.30
MOCC	20	31.27	68.25	42.89
ITCC	20	31.06	6.16	10.28

Table 2. Comparison of results of LDCC, MOC, MOCC and ITCC algorithms on Wikipedia Corpus in terms of Precision, Recall and F-Measure.

5 Conclusion

This paper has introduced a generative model for simultaneously clustering documents and terms. Latent Dirichlet Co-Clustering (LDCC) models each document as a random mixture of *document topics*, where each topic is a distribution over some segments of the text. Each of these segments in the document can be modeled as a mixture of *word topics* where each topic is a distribution over words. Efficient approximate inference techniques based on Markov Chain Monte Carlo method and a Moment-Matching algorithm for empirical Bayes parameter estimation has been proposed. We have reported promising results on two text datasets, a subset of Wikipedia articles and NIPS conference papers. We compare the proposed model with the Latent Dirichlet Allocation (LDA) model in terms of its ability in document modeling and show improved performance in terms of perplexity. We also compare our algorithm for document clustering with several other clustering and co-clustering algorithms and demonstrate improved performance in terms of clustering quality.

Acknowledgements

We are grateful to the following institutions for their financial support: the Natural Sciences and Engineering Research Council of Canada, IT Interactive Services Inc., and GINIus Inc.

References

[1] A. Banerjee, C. Krumpelman, S. Basu, R. Mooney, and J. Ghosh. Model based overlapping clustering. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, August 2005.

[2] D. Blei and J. Lafferty. Correlated topic models. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 147–154. MIT Press, Cambridge, MA, 2006.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[4] W. L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research (JAIR)*, 2:159–225, 1994.

[5] H. Chipman and H. Gu. Interpretable dimension reduction. *Journal of Applied Statistics*, 32(9):969–987, November 2005.

[6] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[7] L. Denoyer and P. Gallinari. The Wikipedia XML Corpus. *SIGIR Forum*, 2006. Last accessed, June 2006.

[8] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD '03: Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 89–98, New York, NY, USA, 2003. ACM Press.

[9] W. R. Gilks. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC, December 1995.

[10] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101 Suppl 1:5228–5235, April 2004.

[11] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 537–544. MIT Press, Cambridge, MA, 2005.

[12] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57, New York, NY, USA, 1999. ACM Press.

[13] M. Keller and S. Bengio. Theme Topic Mixture Model: A Graphical Model for Document Representation. IDIAP-RR 05, IDIAP, 2004.

[14] W. Li and A. Mccallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *ICML '06: 23rd International Conference on Machine Learning*, Pittsburgh, Pennsylvania, USA, June 2006.

[15] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 1(1):24–45, 2004.

[16] T. Minka and J. Laferty. Expectation-propagation for the generative aspect model, 2002.

[17] T. P. Minka. Estimating a Dirichlet distribution. Technical report, MIT, 2000.

[18] M. Newton and A. Raftery. Approximate Bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society. Series B*, 56:3–48, 1994.

[19] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[20] M. Shafiei and E. Milios. Model-based overlapping co-clustering. In *Proceedings of the Fourth Workshop on Text Mining, Sixth SIAM International Conference on Data Mining*, Bethesda, Maryland, April 22 2006.

[21] M. Steyvers and T. Griffiths. Probabilistic topic models. In T. Landauer, D. Mcnamara, S. Dennis, and W. Kintsch, editors, *Latent Semantic Analysis: A Road to Meaning*. Laurence Erlbaum, 2005.

[22] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 2006.

A Gibbs Sampling Derivations

Beginning with the joint distribution $p(\mathbf{w}, \mathbf{z}, \mathbf{y})$, we can take advantage of conjugate priors to simplify the formulae. All symbols are defined in Section 3 and Table 1.

$$
\begin{aligned}
& p(\mathbf{w}, \mathbf{z}, \mathbf{y}) = p(\mathbf{w}|\mathbf{z}, \eta) p(\mathbf{z}|\alpha, \mathbf{y}) p(\mathbf{y}|\delta) \\
= & \int p(\mathbf{w}|\mathbf{z}, \beta) p(\beta|\eta) d\beta \int p(\mathbf{z}|\theta) p(\theta|\alpha, \mathbf{y}) d\theta \int p(\mathbf{y}|\phi) p(\phi|\delta) d\phi \\
= & \int \prod_{d=1}^{M} \prod_{s=1}^{S_d} \prod_{n=1}^{N_s d} p(w_{z_{dsn}}|\beta_{z_{dsn}}) \prod_{l=1}^{L} p(\beta_l|\eta) d\beta \\
& \int \prod_{d=1}^{M} \prod_{s=1}^{S_d} \left(\prod_{n=1}^{N_s d} p(z_{dsn}|\theta_{ds}) p(\theta_{ds}|\alpha, y_{ds}) \right) d\theta \\
& \int \prod_{d=1}^{M} \left(\prod_{s=1}^{S_d} p(y_{ds}|\phi_d) p(\phi_d|\delta) \right) d\phi \\
= & \int \prod_{l=1}^{L} \prod_{v=1}^{V} \beta_{lv}^{n_{lv}} \prod_{l=1}^{L} \left(\frac{\Gamma(\sum_{v=1}^{V} \eta_v)}{\prod_{v=1}^{V} \Gamma(\eta_v)} \prod_{v=1}^{V} \beta_{lv}^{\eta_v - 1} \right) d\beta \\
& \int \prod_{d=1}^{M} \prod_{s=1}^{S_d} \prod_{l=1}^{L} \theta_{dsl}^{n_l^{(ds)}} \prod_{d=1}^{M} \prod_{s=1}^{S_d} \left(\frac{\Gamma(\sum_{l=1}^{L} \alpha_{y_{ds} l})}{\prod_{l=1}^{L} \Gamma(\alpha_{y_{ds} l})} \prod_{l=1}^{L} \theta_{dsl}^{\alpha_{y_{ds} l} - 1} \right) d\theta \\
& \int \prod_{d=1}^{M} \prod_{k=1}^{K} \phi_{dk}^{P_{dk}} \prod_{d=1}^{M} \left(\frac{\Gamma(\sum_{k=1}^{K} \delta_k)}{\prod_{k=1}^{K} \Gamma(\delta_k)} \prod_{k=1}^{K} \phi_{dk}^{\delta_k - 1} \right) d\phi \\
= & \left(\frac{\Gamma(\sum_{v=1}^{V} \eta_v)}{\prod_{v=1}^{V} \Gamma(\eta_v)} \right)^{L} \prod_{l=1}^{L} \frac{\prod_{v=1}^{V} \Gamma(n_{lv} + \eta_v)}{\Gamma(\sum_{v=1}^{V} n_{lv} + \eta_v)} \\
& \prod_{d=1}^{M} \prod_{s=1}^{S_d} \left(\frac{\Gamma(\sum_{l=1}^{L} \alpha_{y_{ds} l})}{\prod_{l=1}^{L} \Gamma(\alpha_{y_{ds} l})} \right) \prod_{d=1}^{M} \prod_{s=1}^{S_d} \frac{\prod_{l=1}^{L} \Gamma(\alpha_{y_{ds} l} + n_l^{(ds)})}{\Gamma(\sum_{l=1}^{L} \alpha_{y_{ds} l} + n_l^{(ds)})} \\
& \left(\frac{\Gamma(\sum_{k=1}^{K} \delta_k)}{\prod_{k=1}^{K} \Gamma(\delta_k)} \right)^{M} \prod_{d=1}^{M} \frac{\prod_{k=1}^{K} \Gamma(\delta_k + P_{dk})}{\Gamma(\sum_{k=1}^{K} \delta_k + P_{dk})}
\end{aligned}
$$

Using the chain rule, we have

$$
\begin{aligned}
p(z_{dsn}|z_{-dsn}, y, w) &= \frac{p(z_{dsn}, w_{dsn}|z_{-dsn}, y, w_{-dsn})}{p(w_{dsn}|z_{-dsn}, y, w_{-dsn})} \\
&\propto \frac{p(z,y,w)}{p(z_{-dsn}, y, w_{-dsn})} \\
&= (\alpha_{y_{ds} z_{dsn}} + n_{z_{dsn}}^{(ds)} - 1) \times \frac{n_{z_{dsn} w_{dsn}} + \eta_{w_{dsn}} - 1}{\sum_{v=1}^{V} n_{z_{dsn} v} + \eta_v - 1}
\end{aligned}
$$

Using chain rule, again we have

$$
\begin{aligned}
p(y_{ds}|z, y_{-ds}, w) &= \frac{p(z_{ds}, y_{ds}, w_{ds}|z_{-ds}, y_{-ds}, w_{-ds})}{p(w_{ds}, z_{ds}|z_{-ds}, y_{-ds}, w_{-ds})} \\
\propto & \frac{p(z,y,w)}{p(z_{-ds}, y_{-ds}, w_{-ds})} \\
= & (\delta_{y_{ds}} + P_{d y_{ds}} - 1) \times \frac{\Gamma(\sum_{l=1}^{L} \alpha_{y_{ds} l})}{\prod_{l=1}^{L} \Gamma(\alpha_{y_{ds} l})} \frac{\prod_{l=1}^{L} \Gamma(\alpha_{y_{ds} l} + n_l^{(ds)})}{\Gamma(\sum_{l=1}^{L} \alpha_{y_{ds} l} + n_l^{(ds)})} \\
= & (\delta_{y_{ds}} + P_{d y_{ds}} - 1) \times \frac{\prod_{l=1}^{L} \prod_{j=0}^{n_l^{(ds)} - 1} (\alpha_{y_{ds} l} + j)}{\prod_{j=0}^{n^{(ds)} - 1} (\sum_{l=1}^{L} \alpha_{y_{ds} l} + j)}
\end{aligned}
$$